HAL
open science

Towards trusted and secure communications in a vehicular environment

Heng Chuan Tan

To cite this version:

Heng Chuan Tan. Towards trusted and secure communications in a vehicular environment. Cryptography and Security [cs.CR]. Télécom ParisTech; Nanyang Technological University (Singapour), 2017. English. NNT : 2017ENST0063 . tel-01794163

HAL Id: tel-01794163

https://pastel.hal.science/tel-01794163v1

Submitted on 17 May 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

2017-ENST-0063

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech and Nanyang Technological University

Spécialité « Informatique et Réseaux »

présentée et soutenue publiquement par

Heng Chuan Tan

le 13 Septembre 2017

Vers des communications de confiance et sécurisées dans un environnement véhiculaire

Directeur de thèse : **Houda Labiod**
Co-encadrement de la thèse : **Ma Maode**

Jury

M. Benoît GELLER, Professor, ENSTA-ParisTech	Examinateur
M. Sherali Zeadally, Associate Professor, University of Kentucky	Examinateur
M. Shen Xuemin, Professor, University of Waterloo, Waterloo	Examinateur
M. Xiao Gaoxi, Associate Professor, Nanyang Technological University	Examinateur
M. Guan Yong Liang, Associate Professor, Nanyang Technological University	Examinateur

TELECOM ParisTech
école de l'Institut Mines-Télécom - membre de ParisTech
46 rue Barrault 75013 Paris - (+33) 1 45 81 77 77 - www.telecom-paristech.fr

T H È S E

TELECOM PARISTECH

Towards Trusted and Secure Communications in a Vehicular Environment

Doctorate Thesis,

Specialty : Computers and Networks

Defended by

Heng Chuan Tan

September 2017, 13th

THESIS COMMITTEE

Benoît	Geller	Professor, ENSTA-ParisTech	Examiner
Sherali	Zeadally	Associate Professor, University of Kentucky	Examiner
Xuemin	Shen	Professor, University of Waterloo, Waterloo	Examiner
Gaoxi	Xiao	Associate Professor, Nanyang Technological University	Examiner
Yong Liang	Guan	Associate Professor, Nanyang Technological University	Examiner

Acknowledgements

I would like to express my sincere gratitude and appreciation to all the people who have helped and inspired me throughout my Ph.D. study. This thesis would not have been possible without the support of many people.

First of all, I would like to thank my thesis supervisor, Prof. Ma Maode for accepting me into the Joint-PhD program with Telecom ParisTech. Under his supervision, he has provided me with valuable ideas and insights to improve my research and is always there when I needed assistance. His high expectations yet cheerful personality have made my Ph.D. journey more pleasant and less stressful. His wide expertise and many years of experience have also helped me to gain a greater understanding of the topic.

Next, I would like to thank my thesis co-supervisor, Prof Houda Labiod for her mentoring during my attachment at Telecom ParisTech between Apr 2014 and Jun 2015. Her constant attention to details and her strict demands on getting things done right have inspired me to analyze problems from different perspectives and challenged me to do a quality research.

I would also like to thank Dr. Zhang Jun for the valuable discussions and encouragement in completing my research, especially his patient in answering my sometimes silly questions.

Special thanks to my colleagues in Infinitus laboratory at NTU and the INFRES department at Telecom ParisTech who had contributed indirectly or directly towards the successful completion of this thesis.

Last but not least, I would like to thank my wife who supported my decision to embark on graduate studies, despite the significant changes it involved in our lives. I thank her for her support and care to finish this thesis. Finally, I want to thank NTU for providing such a wonderful learning environment towards the completion of my Ph.D.

Résumé

Les progrès des technologies de communication numérique et sans fil ont révolutionné le fonctionnement du monde des transports. Les véhicules actuels ne sont plus des véhicules mécaniques traditionnels mais des machines sophistiquées équipées de nombreux dispositifs de communication intelligents tels que des capteurs, un système de positionnement global (GPS) et une puissante unité embarquée (OBU). Grâce à ces appareils intelligents, les véhicules peuvent partager des informations vitales entre eux et interagir intelligemment avec l'infrastructure environnante. Grâce à ces dispositifs intelligents, les véhicules peuvent partager entre eux des informations vitales et interagir intelligemment avec l'infrastructure environnante pour former un réseau de véhicules connectés appelé Véhiculaires Ad Hoc Network (VANET). Avec l'avènement des VANET, de nombreuses applications véhiculaires prometteuses comme le système d'avertissement de collision et les stations de péage électronique peuvent être développées pour améliorer la sécurité routière et améliorer l'efficacité globale de l'écosystème de transport. Avec l'avènement des VANET, la sécurité routière et l'efficacité globale de l'écosystème de transport sont améliorés. De plus, des applications de confort et de divertissement ou bien des centres d'intérêt peuvent être mis en place pour rendre le voyage des usagers de la route plus agréable.

Alors que VANET continue d'évoluer vers la prochaine génération des technologies de transport, les défis techniques tels que l'interopérabilité, l'évolutivité, la gestion de réseau et la sécurité restent à résoudre. En particulier, la sécurité constitue une menace sérieuse en cette ère de connectivité en raison de l'environnement ouvert et de la nature multi-hop des communications. Comme de nombreux autres réseaux sans fil, un attaquant peut lancer des attaques par déni de service (DoS) en inondant les réseaux de communication pour les faire tomber. Un attaquant peut espionner et usurper l'identité d'autres usagers légitimes de la route pour avoir accès à ses données confidentielles. D'autres attaques incluent des modifications du contenu des messages qui pourraient induire en erreur les usagers de la route et causer des dommages catastrophiques pour la vie ou bien des atteintes à la propriété et provoquer la perte de revenus. Pour ces raisons, VANET doit adopter des mesures de sécurité robustes pour protéger ses communications contre les attaquants externes et internes. En conséquence, cela a conduit à une sensibilisation accrue de la part des universités et de l'industrie pour intensifier les efforts de recherche afin d'étudier de

nouvelles mesures de sécurité pour empêcher la divulgation de données et sécuriser le système contre les cyber-attaques.

Le but de cette thèse est d'étudier en détail les challenges de sécurité et de proposer des solutions. L'objectif est de réaliser des communications fiables et sécurisées dans l'environnement véhiculaire. À cette fin, les solutions proposées s'inspirent de deux aspects, à savoir l'approche cryptographique et non cryptographique. Elles sont organisées en six chapitres où chaque chapitre se concentre sur des aspects différents. Le chapitre 1 présente les bases du réseau véhiculaire et discute des défis à relever pour y motiver la recherche. Le chapitre 2 passe en revue les mesures de sécurité de pointe dans les réseaux de véhicules, en particulier celles liées à la gestion de la confiance et à la gestion des clés. La contribution principale commence à partir du chapitre 3 où un modèle de confiance est proposé pour traiter les recommandations de confiance malhonnêtes. Le chapitre 4 s'appuie sur le modèle de confiance du chapitre 3 pour détecter les attaques par modification de paquet afin d'améliorer la précision de l'évaluation de la confiance. Le chapitre 5 présente un cadre de sécurité qui utilise des clés symétriques comme alternative à la cryptographie asymétrique intensive en ressources pour sécuriser les messages de communication. En outre, le cadre de sécurité propose des contre-mesures pour améliorer l'efficacité de l'infrastructure à clé publique (PKI). Enfin, le chapitre 6 conclut la thèse et souligne quelques travaux futurs.

Chapitre 1

Le premier chapitre est introductif et couvre quelques connaissances de base sur les VANET pour jeter les bases d'un examen plus approfondi des problèmes et de la discussion des solutions dans les chapitres suivants. Ce chapitre est organisé en trois parties.

La partie 1 commence par une brève description des technologies en évolution qui permettent à un véhicule de communiquer avec d'autres véhicules et l'environnement. Cette section décrit en outre comment les VANET ont évolué pour devenir des réseaux maillés (V-Mesh) dans le but de devenir partie intégrante d'un système de transport coopératif intelligent (CITS) pour prendre en charge une gamme

plus large de services. Ceci est accompli en configurant les équipements embarqués ou *Roadside Units* (RSU) pour fonctionner dans le mode de réseau maillé sans fil (WMN) qui est une technologie sans fil naissante pour étendre la couverture de réseau d'un VANET pur. Les avantages de l'utilisation de WMN sur la technologie cellulaire sont (1) le déploiement rapide avec des liaisons à moindre coût, (2) la facilité à fournir une couverture dans les zones difficiles à câbler, (3) l'auto-réparation et l'extensibilité du réseau, (4) la grande portée des communications en raison de la transmission multi-sauts, (5) la bande passante plus élevée en raison des sauts plus courts et enfin (6) la meilleure vie de la batterie en raison de transmissions de puissance inférieure.

La partie 2 présente la vue architecturale d'un réseau V-Mesh et décrit les différents types de liens de communication, les fonctionnalités difficiles à traiter et les différents types d'applications. Très brièvement, l'architecture V-Mesh se compose de quatre composants ITS (*Intelligent Transportation System*). Le système central ITS comprend des autorités publiques telles que le système de gestion du trafic (TMS), l'autorité de certification (CA) et les fournisseurs de services (SPs) dont le rôle est de fournir des fonctions de gestion, d'administration et de soutien au système de transport. Dans toute la région. Le système RSU ITS comprend des unités RSU qui sont des dispositifs fixes installés aux feux de circulation, aux portiques ou aux jonctions critiques pour prendre en charge la mise en réseau maillée (IEEE 802.11s) ainsi que la communication inter-véhicules (IEEE 802.11p). Le système ITS du véhicule comprend des véhicules appartenant à des intérêts privés et publics voyageant à des vitesses différentes. Chaque véhicule est équipé d'un OBU (*On-Board Unit ou unité embarquée*) équipé d'une radio 802.11p pour collecter des informations sur la dynamique du véhicule comme la direction du cap et la vitesse. Le système ITS fait référence aux appareils personnels portables tels que les smartphones, les tablettes, les appareils de navigation personnels qui sont transportés par des piétons et des cyclistes dans le réseau. Les utilisateurs de ces appareils obtiennent généralement des services liés aux ITS grâce aux diverses applications installées en utilisant la technologie cellulaire ou Wi-Fi. Pour soutenir les communications véhiculaires, quatre types de liens sont définis : véhicule-infrastructure (V2I), véhicule-véhicule (V2V), véhicule-piéton (V2P), piéton-infrastructure (P2I) et Infrastructure à Infrastructure

(I2I). Comme le réseau V-Mesh est construit à partir d'un VANET, il hérite de nombreuses caractéristiques telles que (1) mobilité variable due au type de véhicule et à l'environnement routier, (2) topologie hautement dynamique avec changements fréquents de topologie, (3) les connexions à courte durée, (4) les communications multi-sauts, (5) contraintes de délai et (6) une taille de réseau non limitée. En termes de scénarios d'application, les applications de réseau V-Mesh peuvent être classées en trois types, à savoir, la sécurité routière, la gestion du trafic et les applications à valeur ajoutée. Les applications de la sécurité routière ont pour fonction d'élargir la perception des usagers de la route afin qu'ils puissent prendre des décisions plus éclairées pour améliorer la sécurité routière. Les applications de gestion du trafic visent à réduire l'impact environnemental du transport en mettant en œuvre des services tels que l'assistance au choix des itinéraires afin de minimiser les risques de congestions. Enfin, les applications à valeur ajoutée sont axées sur la fourniture de services de confort et de commodité tels que des informations sur les points d'intérêt pour les usagers de la route afin de rendre leur voyage plus agréable.

La partie 3 présente deux problèmes qui affectent l'efficacité des communications dans un réseau V-Mesh. Le premier problème est causé par les communications multi-sauts inhérentes où la coopération entre les nœuds est difficile à appliquer pendant le routage. Certains nœuds peuvent supprimer le transfert afin d'économiser leurs ressources. En conséquence de ces mauvais comportements, le réseau souffre d'une dégradation importante du débit. Si des messages importants sont perdus, cela peut amener les usagers de la route à prendre de mauvaises décisions et à créer un chaos dans la circulation ou même des accidents de la route. Une approche pour résoudre le problème de l'égoïsme des nœuds consiste à déployer un modèle de confiance dans le protocole de routage pour aider à distinguer les nœuds bien élevés et les nœuds égoïstes. Le rôle du modèle de confiance consiste à évaluer la fiabilité de chaque nœud de transmission en calculant ses cotes de confiance en fonction des interactions directes et des observations indirectes recueillies auprès des autres membres du réseau. Cependant, le problème de la collecte d'observations indirectes est que le recommandataire peut être malhonnête et mentir sur les cotes de confiance pour influencer la valeur de confiance des autres. À cette fin, le premier objectif est de :

“Concevoir un modèle de confiance qui peut atténuer les effets de fausses recommandations”.

Outre la question des faux positifs, l'utilisation de l'écoute des transmissions pour évaluer la cote de confiance d'un nœud est peu fiable donc imprécise. Un attaquant peut contrôler la puissance de transmission de sorte que le paquet soit reçu mais pas transmis au saut suivant. Un attaquant peut également modifier le contenu d'un paquet avant le transfert qui ne peut pas être détecté si un nœud ne parvient pas à l'écouter. Par conséquent, la cote de confiance basée sur l'écoute est très ambiguë. Le deuxième objectif de cette thèse est de :

“Développer un modèle de confiance pour améliorer le processus d'audition et détecter les modifications de message pendant le transfert de paquet”.

Enfin, le réseau V-Mesh est un système de nœuds qui opère dans un environnement multi-saut très dynamique dans lequel il y a des contraintes de délais pour assurer l'efficacité. L'utilisation d'une PKI est inappropriée car elle implique une latence élevée pour vérifier les certificats numériques et les signatures numériques du nœud. Ce problème est encore compliqué par des caractéristiques comme la déconnexion fréquente où les temps de connexion courts. Cela qui rend le fonctionnement de PKI incapable de prendre en charge les applications ITS critiques, en particulier dans les applications liées à la sécurité. Le dernier objectif est de :

“Développer une approche alternative à PKI qui peut répondre aux exigences de latence sans compromettre la sécurité”.

Chapitre 2

Le deuxième chapitre passe en revue les attaques de sécurité dans un réseau V-Mesh et leur impact sur les objectifs de sécurité. Deux défis clés dans le modèle de confiance et le système de gestion des clés sont identifiés et revus. Dans le premier, une brève introduction sur les modèles de confiance est présentée. Elle est suivie d'une description de la façon dont les modèles de confiance existants traitent et regroupent les fiducies ensemble. Ce chapitre traite également de plusieurs modèles basés sur l'écoute et explique comment les attaques liées à l'écoute peuvent altérer

le processus d'évaluation de la confiance. Les travaux connexes pour surmonter les complexités du déploiement de PKI et leurs limites sont discutés.

La partie 1 commence par une classification des attaques traditionnelles. En général, les attaques peuvent être classées en attaques externes et internes. Les attaques externes sont menées par des attaquants qui n'ont pas le droit d'accéder aux services réseau. Ils ne peuvent que compromettre le réseau en écoutant le trafic ou en inondant le réseau de faux messages. Par conséquent, les attaques externes peuvent être facilement évitées en utilisant des techniques cryptographiques classiques telles que le cryptage, le contrôle d'accès et le pare-feu. D'un autre côté, les attaques internes sont lancées par des initiés qui possèdent des clés cryptographiques valides et qui sont familiers avec les politiques et les procédures du système. Puisque ces attaquants font partie de l'intimité du réseau, ils sont plus sévères et difficiles à détecter que les attaques externes. Outre les attaques externes et internes, les attaques peuvent être classées en attaques passives et actives. Les attaques passives visent à obtenir des informations ou des données sensibles en surveillant le canal de communication tandis que les attaques actives impliquent l'altération, l'injection, la relecture et la suppression de messages dans le réseau. Enfin, cinq types d'objectifs de sécurité, que les différents types d'attaques tentent de saper, sont discutés. Ce sont la confidentialité, l'intégrité, la disponibilité, l'authentification et la non-répudiation. Ces objectifs de sécurité constituent la base de la communication. La compréhension de ces objectifs est essentielle à la conception de solutions de sécurité robustes et efficaces pour les réseaux V-Mesh.

La partie 2 passe en revue les défis rencontrés par les réseaux V-Mesh et les résume en cinq grandes catégories : gestion des clés, routage, contrôle d'accès, authentification et confidentialité. Selon l'examen, le routage et la gestion des clés sont les deux principaux défis.

Il est important de sécuriser le routage V-Mesh car de nombreuses applications et services véhiculaires en dépendent pour un transfert de données efficace. De plus, la plupart des protocoles de routage sont conçus sans aucune considération de sécurité, c'est-à-dire que tous les nœuds participants sont considérés comme fiables et prêts à coopérer les uns avec les autres. Pour ces raisons, le routage est une surface d'attaque

potentielle que les attaquants peuvent exploiter pour perturber les communications véhiculaires. Un nœud égoïste peut déclencher une attaque *blackhole* ou *grayhole* en refusant de transmettre les paquets de données. Plus spécifiquement, un nœud égoïste déclare avoir une route valide vers la destination en envoyant immédiatement un paquet RREP (*Route Reply*) au nœud source. Lorsque le nœud source reçoit la première copie du paquet RREP, il suppose que la découverte de l'itinéraire est terminée et ignore tous les autres RREP des autres nœuds. Cela induit ainsi le nœud source en erreur en sélectionnant le chemin contenant le nœud égoïste en tant que redirecteur. Par la suite, le nœud égoïste consomme tout le trafic sans le transmettre. Si un nœud égoïste supprime tous les paquets de données, ce type d'attaque par largage est appelé attaque *blackhole*. Si seulement une partie des paquets de données est abandonnée, elle s'appelle une attaque *grayhole*. Un autre problème de routage apparaît lorsqu'un nœud peut injecter malicieusement de fausses informations ou modifier des paquets de données existants pendant le processus d'acheminement. C'est ce que l'on appelle souvent les attaques par modification de paquet pour violer l'intégrité des données reçues. Dans le pire des cas, si certaines parties du message qui sont utilisées à des fins de suivi sont effacées délibérément, les modifications peuvent également affecter la propriété de non-répudiation qui est difficile à résoudre.

Dans le domaine de la gestion des clés, il est difficile de concevoir un cadre de gestion des clés sûr et efficace pour créer, distribuer et protéger les clés cryptographiques pour des communications sécurisées. Cela est dû au degré élevé de mobilité des nœuds, aux fréquentes déconnexions et aux temps de connexion courts, en particulier lorsque la PKI est utilisée. Lorsque des certificats sont utilisés, le destinataire doit (1) vérifier la date d'expiration du certificat, (2) télécharger la liste de révocation des certificats (CRL), (3) vérifier la validité du certificat par rapport à la CRL, (4) vérifier les CA signature sur le certificat signé reçu, et (5) récupérer la clé publique de l'expéditeur et enfin vérifier le message de l'expéditeur. Ceci entraîne une latence élevée qui ne répond pas aux exigences des applications liées à la sécurité. De plus, les clés émises dans le certificat peuvent ne pas être récentes car la CRL n'est pas publiée en temps réel mais périodiquement. Par conséquent, un destinataire peut utiliser un certificat périmé qui a été révoqué.

La partie 3 passe en revue les bases d'un modèle de confiance et examine les travaux

antérieurs sur la gestion de la confiance. La motivation de l'utilisation du modèle de confiance est que les primitives cryptographiques traditionnelles ne sont pas capables d'empêcher les attaques dues à des comportements égoïstes, c'est-à-dire des attaques de *blackhole* et *grayhole*. Pour résoudre ces attaques, il faut compter sur des modèles basés sur la confiance pour évaluer le comportement de transfert des nœuds. Cependant, les différentes étapes (surveillance, agrégation, décision) d'un modèle de confiance peuvent également être corrompues. À cette fin, cette partie passe en revue chaque composante du modèle de confiance et en discute les vulnérabilités.

Dans l'étape d'agrégation, l'écoute est couramment utilisée pour évaluer le comportement d'acheminement d'un nœud en écoutant la transmission du nœud en aval. Si le nombre de paquets abandonnés dépasse un certain seuil, le nœud en aval est considéré comme égoïste en réacheminement. Cependant, l'écoute peut ne pas être capable d'écouter chaque transmission en raison de l'évanouissement inévitable des canaux, des collisions et des interférences. L'écoute est également vulnérable aux mauvais comportements dans lesquels un nœud malveillant peut lancer une attaque de puissance de transmission et de modification de paquet limitée. Dans une attaque de puissance de transmission limitée, un attaquant contrôle la puissance de transmission de sorte que la transmission de paquets est écoutée sur le saut précédent mais est trop faible pour être reçue par le saut suivant. En conséquence, le nœud précédent détecte toujours l'attaquant en tant que nœud de confiance et inclut l'attaquant dans le chemin de routage. Dans une attaque par modification de paquet, un attaquant modifie le message avant de le transmettre et obtient toujours une note positive du processus d'évaluation de la confiance. Par conséquent, les attaquants de modification de paquet seront toujours évalués comme nœud de confiance pour effectuer le transfert de paquet.

Un examen des travaux connexes révèle que bon nombre des modèles de confiance existants n'ont pas entièrement résolu les problèmes de sur-déclaration inexacte, de puissance de transmission limitée et d'attaques par modification de paquet. Une tentative a été faite dans la littérature pour contrer les imperfections de l'écoute. L'idée est d'exploiter plus de nœuds dans le voisinage pour rapporter les résultats de l'écoute et utiliser la théorie de Dempster Shafer (DST) pour les agréger. Cependant, cette approche n'est pas efficace si le réseau est clairsemé. Pour traiter les attaques par modification de paquet, de nombreux modèles de confiance existants

proposent d'utiliser le même mécanisme de sur-écoute pour vérifier l'intégrité des paquets transférés. Cependant, cette approche est inefficace parce que l'écoute en sans-fil est très sensible, ce qui provoque parfois un dysfonctionnement. En termes d'attaques de puissance de transmission limitées, une méthode consiste à demander au nœud à deux sauts de renvoyer le nombre de paquets reçus pour confirmer les observations de sur-écoute du nœud amont. Cependant, le modèle de confiance proposé qui adopte cette méthode n'est pas optimisé car la détection de nœud égoïste est effectuée par le nœud de destination qui est plus lent. De plus, il ne s'appuie pas sur les techniques de confiance indirect dont on a prouvé dans la littérature qu'elles améliorent le temps de détection.

Dans l'étape d'agrégation de confiance, les observations directes et indirectes sont combinées pour former les valeurs de confiance. L'objectif est d'améliorer la précision des évaluations de confiance ainsi que le temps de détection des nœuds égoïstes dans le réseau. Cependant, cette étape est sensible aux attaques de *badmouthing* et de bourrage des urnes (*ballot stuffing*) par les recommandeurs. Dans le premier, le recommandeur tente intentionnellement de ruiner intentionnellement la confiance d'un bon nœud en fournissant de mauvaises recommandations afin de diminuer les chances que ce nœud soit sélectionné pour le service. Il en résulte une liste noire de bons nœuds provoquant ainsi une partition dans le réseau qui va diminuer le débit et la fiabilité globale du réseau. Le recommandeur malveillant renforce la confiance d'un autre nœud malveillant en fournissant de bonnes recommandations afin d'augmenter les chances que ce nœud malveillant soit sélectionné en tant que redirecteur. Une fois sélectionné, le nœud malveillant peut mener d'autres attaques telles que des attaques de *blackhole* ou *grayhole*.

Diverses techniques d'agrégation de confiance ont été proposées. Elles peuvent être généralisées en quatre types. Dans la méthode de regroupement d'opinion linéaire, la confiance indirecte est pondérée avant d'être combinée linéairement avec la confiance directe. La fonction de pondération peut être configurée en tant que niveau de confiance du recommandeur ou elle peut être définie arbitrairement pour tenir compte des effets d'une déclaration erronée. La seconde technique d'agrégation de confiance est appelée le modèle probabiliste basé sur l'entropie. Dans ce modèle, la propagation de confiance est modélisée de manière probabiliste selon un canal symétrique binaire. La probabilité dérivée est ensuite convertie en valeur de confiance

en utilisant la fonction binaire d'entropie de Shannon. La troisième technique d'agrégation est la logique subjective qui traite la connaissance imparfaite comme une incertitude et la quantifie dans les relations de confiance. Par conséquent, cela conduit aux concepts de croyance, d'incrédulité et d'incertitude qui forment la base de la logique subjective. Puisque la logique subjective peut expliquer l'incertitude, elle améliore la clarté et l'expressivité d'une opinion par rapport à la logique probabiliste traditionnelle. La logique subjective définit en outre deux opérations pour gérer les propagations de confiance, à savoir l'opérateur d'actualisation et l'opérateur de consensus. La dernière technique d'agrégation de confiance est la méthode d'analyse de régression qui est un processus statistique itératif pour prédire la fiabilité d'un nœud en fonction de certains facteurs opérationnels et environnementaux. Le but est d'estimer les coefficients de régression de telle sorte que les probabilités logarithmiques d'un nœud en fournissant un service satisfaisant soient supérieures à 0. Les coefficients de régression sont ensuite remplacés dans la fonction logistique pour calculer la valeur de confiance du nœud. Pour lutter contre les recommandations malhonnêtes, la distribution logistique est remplacée par un bruit blanc dans la distribution afin que les recommandations de variance plus élevées soient correctement pondérées.

En dehors de ces techniques d'agrégation de confiance, des contraintes supplémentaires sont nécessaires pour traiter les recommandations malhonnêtes. Tout d'abord, un filtrage basé sur un seuil est mis en œuvre pour filtrer les mauvaises recommandations si le niveau de confiance du recommandeur est inférieur à un certain seuil ou si l'écart entre la confiance directe et la confiance indirecte est supérieur à un certain seuil. La deuxième approche consiste à autoriser uniquement la propagation des recommandations positives afin que les recommandations négatives ne puissent pas influencer l'agrégation de la confiance. Cependant, aucune des techniques d'agrégation de confiance qui ont utilisé ces contraintes supplémentaires, à savoir le filtrage à seuil, la recommandation positive uniquement ou les méthodes statistiques, n'est efficace pour atténuer les recommandations malhonnêtes. Par exemple, le filtrage basé sur un seuil est difficile à déterminer. Si le seuil est trop élevé, de nombreuses recommandations seront exclues, ce qui va à l'encontre du but de la collecte de recommandations. Si le seuil est trop bas, les fausses recommandations se propageront dans le réseau. Un recommandeur malveillant peut également se comporter

correctement pour contourner la vérification du seuil mais émet ensuite de fausses recommandations pour ruiner la réputation d'un nœud. D'un autre côté, l'approche des recommandations uniquement positives limite le caractère pratique des systèmes de confiance et les rend moins adaptables aux changements dans le réseau. Pour l'analyse de régression, le processus est long à converger.

La partie 4 présente un examen des solutions de sécurité visant à réduire les coûts élevés (en bande passante et en temps) du déploiement d'une infrastructure à clé publique qui peuvent être classées en solutions pour la cryptographie symétrique et pour la cryptographie asymétrique.

Dans le domaine de la cryptographie symétrique, plusieurs schémas de distribution de clés ont été proposés pour échanger des clés de manière efficace et sécurisée. Ce sont le partage de clé probabiliste, la distribution de clé polynomiale, la distribution de clé matricielle, la distribution de clé basée sur une grille, Diffie-Hellman (DH) avec chaîne de hachage et les schémas basés sur l'extraction de signaux. Très brièvement, le partage de clé probabiliste établit une clé probabiliste basée sur des clés communes dans les anneaux de clés donnés. La distribution des clés basée sur le polynôme, la distribution des clés basée sur la matrice et la distribution des clés basée sur la grille impliquent toutes des équations préchargées ou une matrice à résoudre pour une clé. L'approche DH utilise une chaîne de hachage pour générer de nombreuses clés de session après l'établissement de la clé DH. Enfin dans l'extraction de signal, la clé est établie en extrayant des bits communs dans les valeurs *Received Signal Strength* (RSS). Dans tous les schémas mentionnés ci-dessus, l'échange d'informations de clé est nécessaire pour dériver les clés communes, ce qui signifie qu'un adversaire peut envoyer beaucoup de faux messages pour des vérifications, ce qui les rend vulnérables aux attaques DoS. De plus, de nombreux schémas de distribution de clés tels que le partage probabiliste, la clé basée sur la grille et l'approche DH sont vulnérables aux attaques de capture de nœud car un adversaire peut physiquement compromettre un nœud pour extraire les clés de préchargement.

En termes d'optimisation de la cryptographie asymétrique, notamment avec PKI, plusieurs approches ont été proposées. Une approche est le schéma PKR (*Public Key Regime*) dans lequel un RSU est attribué à un répertoire de clé publique contenant la clé publique et la paire d'ID de chaque véhicule du réseau. Ainsi, le débit de trans-

mission et la latence sont réduits car le téléchargement de CRL et les processus de vérification de certificat qui prennent du temps sont éliminés. Une autre approche consiste à utiliser la cryptographie basée sur l'identité (IBC) lorsque l'identité de l'utilisateur est la clé publique pour éliminer la nécessité de stocker, d'émettre et de vérifier les certificats. Une autre approche propose l'utilisation d'unités RSU pour émettre et révoquer des certificats de manière ciblée tandis que d'autres utilisent un filtre bloom pour réduire la taille de la CRL et effacer l'encodage pour encoder la liste CRL en plusieurs éléments auto-vérifiables. Un protocole *Expedite Message Authentication Protocol* (EMAP) a également été proposé pour surmonter le long délai de vérification de l'état de révocation d'un certificat à l'aide d'une liste CRL. On vérifie le HMAC d'abord avant que le certificat numérique et la signature soient vérifiés.

Chapitre 3

Ce chapitre présente le modèle de Dempster Shafer-Trust (DS-Trust) pour évaluer la fiabilité des nœuds. L'objectif est de fournir à chaque nœud intermédiaire une évaluation de confiance précise du transitaire pour effectuer le routage. Il est conçu pour aider le protocole de routage sous-jacent à détecter et isoler les nœuds égoïstes sur le chemin afin d'augmenter le débit. Il est également conçu pour détecter les nœuds malicieux qui signalent des évaluations de confiance incorrectes telles que la rétrogradation ou la promotion de la réputation des autres membres du réseau. La nouveauté du modèle DS-Trust est l'utilisation de la DST (*Dempster–Shafer Theory*) pour caractériser la modélisation de confiance et l'agrégation de confiance. Grâce à l'utilisation de la DST, les recommandations contradictoires sont réduites de manière appropriée. Il n'est pas nécessaire d'effectuer une conception de filtrage complexe et complexe basée sur des seuils, ce qui peut parfois faussement éliminer de véritables recommandations. Pour ces raisons, la DST est particulièrement bien adaptée pour établir des relations de confiance, en particulier lorsque les opinions de confiance des autres pairs du réseau sont contradictoires en raison de comportements erronés. Par conséquent, le modèle DST est considérée comme une bonne alternative à la théorie des probabilités traditionnelle en termes de clarté et d'expressivité. Le chapitre est subdivisé en cinq parties.

La partie 1 traite du type d'attaques de sécurité que le modèle DS-Trust peut résoudre. Plus précisément, le modèle DS-Trust peut détecter le largage de paquets générés par des nœuds égoïstes tels que les attaques *blackhole* et *grayhole*. La principale différence entre les deux est que les attaquants *blackhole* perdent 100% des paquets de données qu'ils reçoivent, tandis que les attaquants *grayhole* abandonnent les paquets en fonction d'une distribution de probabilité aléatoire. Pour cette raison, les attaques *grayhole* sont difficiles à détecter car la perte de données peut être perçue comme une perte due à la congestion du réseau, phénomène naturel dans tous les réseaux sans fil. En conséquence, les attaques *grayhole* peuvent ne pas être détectées pendant une longue période causant plus de dégâts que les attaques *blackhole*. L'impact des attaques par largage de paquets est une réduction du débit et de la disponibilité du service. En plus des attaques de largage de paquets, le modèle DS-Trust peut atténuer les effets néfastes des attaques de *badmouthing* et de bourrage d'urnes. Les attaques malfaisantes se réfèrent à des nœuds malveillants fournissant des recommandations de mauvaise confiance pour encadrer les bons pairs afin de diminuer leur cote de confiance. Dans le cas d'une attaque par bourrage d'urnes, un nœud malveillant fournit de bonnes recommandations de confiance pour augmenter les cotes de confiance d'un pair malveillant afin d'éluder la détection ou d'augmenter les chances d'être sélectionné comme transitaire.

La partie 2 passe en revue les bases de la DST, qui est une théorie mathématique de la preuve basée sur des fonctions de croyance et un raisonnement plausible. Contrairement à la théorie des probabilités traditionnelle, DST permet une représentation explicite de l'incertitude et ne satisfait pas la règle d'additivité de la probabilité, c'est-à-dire que l'on n'a pas $P(A) + P(\bar{A}) = 1$. Pour clarifier cela, supposons que le nœud de transfert est digne de confiance avec une croyance de $P(A) = 0.8$. Si la théorie des probabilités traditionnelle est utilisée, la probabilité restante de 0.2 doit être attribuée à l'état non approuvé, c'est-à-dire $P(\bar{A})$ pour satisfaire à la règle d'additivité de la probabilité. En revanche, la DST classe la croyance restante en tant qu'incertitude, ce qui signifie qu'un nœud peut alors être considéré comme fiable ou non fiable. Cet argument est raisonnable car il n'y a aucune preuve pour justifier que le nœud ne soit pas digne de confiance. En introduisant l'incertitude dans le modèle

de confiance, une fausse détection causée par une accusation prématurée peut être évitée. La DST peut également gérer des preuves contradictoires résultant d'observations imparfaites et de mauvaise conduite initié par les différents utilisateurs du réseau. Pour résoudre de tels conflits et incohérences, la DST utilise un opérateur de fusion appelé règle de combinaison de Dempster pour combiner des preuves où l'accord entre des preuves multiples est renforcé par un facteur de normalisation.

Afin d'utiliser la DST pour l'agrégation de confiance, trois fonctions sont définies. L'ensemble de puissance d'un ensemble Θ est défini comme l'ensemble de tous les sous-ensembles de Θ y compris l'ensemble vide et Θ lui-même. Suivant cette définition, la première fonction de la DST est appelée l'assignation de probabilité de base (*bpa*) qui est une mesure du poids assigné à chaque sous-ensemble de l'ensemble de puissance. La deuxième fonction est liée aux *bpas* et s'appelle fonction de croyance et de vraisemblance. Plus formellement, la fonction de croyance est définie comme la somme de tous les *bpas* qui supportent directement une proposition. D'autre part, la fonction de vraisemblance englobe tous les *bpas* qui supportent partiellement ou totalement une hypothèse, c'est-à-dire que tous les *bpas* dont l'interaction avec l'hypothèse n'est pas vide sont considérés comme des croyances potentielles. Cela peut être interprété comme une croyance optimiste. La dernière fonction définit la règle de combinaison de Dempster qui combine plusieurs éléments de preuve ensemble. L'idée principale sous-jacente à la règle combinatoire est de normaliser la somme des produits de toutes les interactions de *bpas* provenant de différents éléments de preuve pour arriver à une croyance commune dans une hypothèse particulière.

La partie 3 décrit le modèle de confiance de la DST qui se compose de cinq modules, à savoir un module de surveillance, un module de rétroaction, un module de corrélation, un module de fusion et un module de décision. La fonction du module de surveillance est de mesurer la confiance directe des autres nœuds du réseau. Il utilise le mécanisme d'écoute pour surveiller le comportement d'acheminement du prochain bond. Chaque nœud conserve une trace du nombre de paquets envoyés et du nombre de paquets écoutés pour calculer la probabilité de réacheminement. La fonction d'entropie binaire de Shannon est ensuite utilisée pour convertir la probabilité d'acheminement en une métrique de confiance directe. L'utilisation de la fonction d'entropie est de prendre en compte la quantité d'incertitude dans la pro-

babilité d'acheminement causée par l'écoute imparfaite. À l'aide de la fonction d'entropie binaire, l'approbation directe est mappée à l'intervalle [0,1] où une valeur de confiance de 0 signifie qu'un nœud n'est pas approuvé et une valeur 1 indique qu'un nœud est entièrement de confiance.

Le module de rétroaction sollicite des recommandations de confiance provenant d'autres nœuds du réseau pour calculer la confiance indirecte. Fondamentalement, chaque nœud diffuse un message à la fin de chaque intervalle de surveillance de confiance pour demander la confiance directe du nœud qui l'intéresse. Lorsque le nœud demandeur reçoit la valeur de confiance directe sous la forme de recommandations, il pèse les recommandations reçues et l'opinion de confiance du conseiller pour calculer la confiance indirecte. Ensuite, le module de corrélation utilise le test de dissimilarité pour quantifier la dissymétrie entre l'enregistrement de confiance direct et les valeurs de confiance indirectes reçues. La dissymétrie est exprimée comme la normalisation de la différence absolue entre les deux valeurs de confiance. Si l'écart est important, cela implique que les deux valeurs de confiance sont en conflit et indique une incertitude. Le résultat du module de corrélation est ensuite transmis au module de fusion où les valeurs de confiance directe et de confiance indirecte sont traitées avant d'être agrégées à l'aide de la règle de combinaison de Dempster.

Ensuite, le module de fusion et le principe de l'agrégation de la DST sont expliqués comme suit. Tout d'abord, un nœud est classé en deux états : approuvé T et non approuvé $\bar{T}$ pour représenter le cadre de discernement sous le raisonnement DST. Les propositions dans le cadre du discernement sont mutuellement exclusives et exhaustives. Sur cette base, l'ensemble de puissance est défini, ce qui étend le cadre de discernement pour inclure tous les sous-ensembles de propositions et l'ensemble vide ϕ c'est-à-dire, ensemble de puissance $=[(T),(\bar{T}),(T,\bar{T}),\phi]$. L'ensemble noté $(T,\bar{T})$ représente l'incertitude, ce qui signifie que le nœud peut être de confiance ou pas. L'attribution de *bpa* à chaque sous-ensemble du groupe de puissance est ensuite déterminée en fonction de certaines règles de classification. Si la valeur de confiance directe est supérieure ou égale à un seuil de détection de confiance, une valeur égale à la valeur de confiance directe sera affectée à l'état sécurisé et la croyance résiduelle est affectée à l'état d'incertitude, c'est-à-dire $(T,\bar{T})$. Pour la classification de la confiance indirecte, le résultat du module de corrélation est utilisé. Fondamentalement, la valeur de dissymétrie est affectée à l'état d'incertitude. Selon la valeur de

la confiance indirecte, la croyance restante est affectée en conséquence. En d'autres termes, si la valeur de confiance indirecte est supérieure ou égale au seuil de détection de confiance, l'état de confiance est attribué à une valeur équivalente à 1 moins la dissymétrie. Ceci est en accord avec les principes de la DST. Deux enregistrements de confiance indiquent l'incertitude. À mesure que la déviation entre la confiance directe et indirecte augmente, la croyance relative à l'événement incertain devrait augmenter en conséquence. Une fois que les *bpas* sont mis à jour pour la confiance directe et indirecte, la règle de combinaison de Dempster est appliquée pour fusionner les preuves de confiance ensemble et le résultat est envoyé au module de décision.

Enfin, le module de décision prend la valeur de confiance agrégée en entrée et décide du cours de l'action lorsque la valeur est inférieure au seuil de détection de confiance. Par exemple, le nœud d'évaluation diffuse une liste noire dans tout le réseau pour informer les autres nœuds de le bloquer de toutes communications futures. Des informations sur le nœud mal géré sont également envoyées au protocole de routage sous-jacent qui déclenchera le protocole de routage pour envoyer un message *Route Error* (RERR) pour notifier le nœud source. Par la suite, une nouvelle découverte d'itinéraire sera initiée pour trouver un chemin libre de nœuds égoïstes.

La partie 4 se concentre sur l'évaluation de la performance du modèle DS-Trust. Par des simulations numériques, les changements de la valeur de confiance en terme du nombre d'attaquants de *badmouthing* et de bourrage d'urnes sont analysés. Les résultats sont ensuite comparés avec d'autres techniques d'agrégation de confiance existantes telles que le regroupement d'opinions linéaire, le modèle de probabilité basé sur l'entropie, la logique subjective et l'analyse de régression. L'objectif est de démontrer que la DST peut atténuer les attaques de *badmouthing* et de bourrage d'urnes plus efficacement que les techniques existantes. Dans la configuration expérimentale, le nœud d'évaluation collecte des recommandations de confiance pour évaluer le niveau de confiance du nœud cible. Il est supposé que le nœud d'évaluation a la même valeur de confiance directe pour chaque recommandeur, de sorte que les changements observés dans les résultats d'agrégation de confiance sont dus à la différence entre les valeurs de recommandation. Pour simuler des attaques de *badmouthing*, chaque recommandeur est configuré pour envoyer une cote de confiance

faible de 0.1. Dans le cas d'une attaque par bourrage d'urnes, chaque recommandeur est configuré pour renvoyer une cote de confiance de 0.9.

Les résultats montrent que le modèle DS-Trust peut empêcher jusqu'à 10 attaquants malfaisants de réduire avec succès la valeur de confiance agrégée en dessous du seuil de détection de confiance de 0.5. Cette amélioration est due au traitement de l'incertitude dans les modules de corrélation et de fusion. Plus précisément, la dissymétrie des deux valeurs de confiance calculées dans le module de corrélation est classée comme incertitude dans le module de fusion pour représenter des états fiables ou non fiables. Par la suite, lorsque la règle de combinaison de Dempster est appliquée pour combiner la confiance directe et la confiance indirecte, l'incertitude est absorbée dans le processus d'agrégation qui amplifie la croyance que le nœud est digne de confiance. Par conséquent, le modèle DS-Trust a une perte de confiance plus lente par rapport aux autres systèmes d'analyse comparative. Dans le cas d'attaques par bourrage d'urnes, des résultats similaires ont été observés. Le modèle DS-Trust est capable de tolérer jusqu'à 10 attaquants avant de succomber aux effets de fausses recommandations alors que le reste des systèmes ne peut tolérer que 2 attaquants. Ces résultats montrent que la règle de combinaison de Dempster proposée peut efficacement atténuer l'impact des attaques de *badmouthing* et bourrage d'urnes. Ensuite, les performances du modèle DS-Trust sont analysées lors d'une attaque *blackhole* et *grayhole*. Le modèle DS-Trust est intégré dans le protocole de vecteur de distance ad hoc à la demande (AODV) et est testé sur deux architectures de réseau V-Mesh différentes. Le premier est un réseau statique basé sur l'infrastructure qui imite le routage dans le réseau maillé des réseaux V-Mesh. La deuxième architecture est un environnement hybride composé de nœuds statiques et mobiles. Dans ce dernier, le nœud mobile peut être un piéton, un cycliste ou un véhicule qui communique avec une RSU statique située dans le réseau principal de maillage. Les simulations ont été effectuées à l'aide du Network Simulator NS-3 (v3.20) et les résultats ont été comparés à l'AODV de référence et à une variante du modèle DS-Trust appelée DS-Trust (sans les recommandations).

Dans le réseau V-Mesh basé sur l'infrastructure sous attaque blackhole, les résultats montrent que le modèle DS-Trust peut améliorer de 30% et 15% le taux de livraison des paquets par rapport à l'AODV de référence et au DS-Trust (sans recommanda-

tion) modèle. En termes de surcharge de routage normalisée, le modèle DS-Trust est plus élevé que le modèle DS-Trust (sans recommandation) en raison de la propagation des recommandations d'approbation. Le modèle DS-Trust présente également un surdébit de routage normalisé supérieur à celui de l'AODV de référence en raison des paquets de contrôle supplémentaires envoyés par le modèle de confiance, notamment l'échange périodique d'informations de confiance, la diffusion de messages de contrôle et la réinitialisation de nouveaux les découvertes d'itinéraires lorsque des nœuds *blackhole* sont détectés.

Deuxièmement, la performance de débit du modèle DS-Trust sous les attaques *blackhole* est étudiée. Les résultats montrent que le modèle DS-Trust peut atteindre un débit supérieur à celui de la DS-Trust (sans recommandation) alors que le taux d'application passe de 16384bps à 65536bps. Au-delà du taux d'application de 65536bps, il n'y a pas d'amélioration des performances entre les deux modèles. Cela est dû à la liste noire des faux positifs situés à proximité du nœud source qui ne génère aucun chemin d'acheminement disponible pour envoyer les paquets à la destination.

Dans le réseau V-Mesh basé sur l'infrastructure sous attaque greyhole, le taux de livraison de paquets du modèle DS-Trust est presque le même que celui de l'AODV de référence lorsque la probabilité de perte sélective est comprise entre 0 et 0.4. Lorsque le taux de perte augmente de 0.4 à 0.5, le taux de livraison de paquets commence à augmenter en moyenne d'environ 80%. L'amélioration est d'environ 28% et 10% supérieure à celle de l'AODV de référence et de la DS-Trust (sans les recommandations). La principale explication à cela est que la note de confiance dans le modèle DS-Trust est calculée en utilisant des taux de confiances directes et indirectes. Au fur et à mesure que d'autres recommandations deviendront disponibles, l'évaluation de la confiance sera plus précise, ce qui conduira à une détection plus rapide des nœuds *greyhole* et à une amélioration du taux de livraison des paquets. En termes de surcharge de routage normalisée, le surdébit est beaucoup plus élevé que le modèle AODV de base et le modèle DS-Trust (sans recommendations). Cependant, le surdébit n'augmente que lorsque le taux d'abandon se situe autour du seuil de détection d'approbation de 0.5, moment auquel le modèle d'approbation commence à générer et à diffuser des paquets de contrôle pour informer les autres nœuds.

Dans le réseau hybride V-Mesh sous attaque blackhole, le modèle DS-Trust est capable d'améliorer le taux de livraison de paquets de 14% et 10% respectivement par rapport au modèle AODV de base et DS-Trust (sans recommandations). Cependant, le taux de livraison de paquets et le gain de performance sont significativement plus bas par rapport aux taux obtenus dans le cas de l'architecture V-Mesh basée sur l'infrastructure car il est plus difficile de maintenir une liaison stable en raison de la mobilité élevée des nœuds. En termes de surcharge de routage normalisée, les performances du modèle DS-Trust sont beaucoup plus faibles que celles de l'AODV de base. Une raison est que lorsqu'un chemin sécurisé est trouvé, il est peu probable qu'il change à moins qu'un nœud ne se trouve hors de portée de transmission. Même si le lien est interrompu en raison de la mobilité, la liste des nœuds *blackhole* a déjà été diffusée aux autres nœuds voisins. Par conséquent, le protocole de routage les évite, ce qui réduit le nombre de nouvelles découvertes de routage et réduit le surdébit de routage normalisé.

Les performances du débit du réseau V-Mesh hybride sont également analysées, le modèle DS-Trust étant capable de maintenir un débit élevé. L'amélioration du débit de la solution DS-Trust est d'environ 25% supérieure à celle de l'AODV de référence et du modèle DS-Trust (sans recommandations).

Dans le réseau V-Mesh hybride sous attaque grayhole, des résultats similaires au réseau V-Mesh basé sur l'infrastructure sont observés. Le taux de livraison de paquets du modèle DS-Trust ne s'améliore que lorsque les attaquants *grayhole* ont un taux de chute de 50% ou plus, qui est le seuil de détection dans le modèle. Cependant, l'amélioration n'est que de 10% supérieure à l'AODV de base, contre 28% dans l'architecture statique. L'amélioration la plus faible est due à la mobilité des nœuds. Au fur et à mesure que les nœuds se déplacent, les liens deviennent relativement instables et peu fiables, ce qui entraîne une perte accrue de paquets. Dans le cas d'un surdébit de routage normalisé, le modèle DS-Trust est plus élevé que le surdébit du modèle DS-Trust (sans les recommandations). Cela est dû à la diffusion des recommandations de confiance. Toutefois, le temps de routage du modèle DS-Trust est inférieur à celui de l'AODV de référence car il permet d'éliminer les nœuds non

fiables pour le routage et de réduire le nombre de découvertes d'itinéraires.

La partie 5 conclut le chapitre en rappelant deux propriétés intéressantes de la DST qui la rendent attrayante pour l'agrégation de confiance. Sur la base de ces propriétés, un modèle de confiance appelé DS-Trust est proposé pour atténuer les attaques de *badmouthing* et de bourrage des urnes. Comme décrit, le modèle DS-Trust proposé se compose de cinq modules : un module de surveillance, un module de rétroaction, un module de corrélation, un module de fusion et un module de décision. Les résultats montrent que DS-Trust peut gérer des informations de confiance hautement trompeuses et résister efficacement aux attaques de *badmouthing* et de bourrage des urnes, notamment par rapport au pool d'opinion linéaire, au modèle logique subjectif, au modèle probabiliste basé sur l'entropie et aux méthodes d'analyse de régression. Le modèle DS-Trust est également testé sur deux architectures de réseau V-Mesh différentes en utilisant NS-3 pour démontrer qu'il résiste aux attaques de largage de paquets et qu'il est capable de récupérer à partir des attaques *blackhole* et *grayhole*.

Chapitre 4

Ce chapitre fournit un modèle de confiance plus complet qui fournit une sécurité supplémentaire contre d'autres attaques liées à la confiance. Comme nous l'avons vu au chapitre 2, la sur-écoute est utilisée pour évaluer le comportement de transfert d'un nœud pendant le routage. Cependant, il est bien connu que l'écoute est vulnérable aux modifications de paquets et aux attaques de puissance de transmission limitées. À cette fin, un modèle de confiance appelé *Merkle Tree-Based with Reinforced Overhearing* (MeTRO) est proposé qui utilise trois techniques pour étendre le travail du modèle DS-Trust.

Tout d'abord, un mécanisme d'authentification basé sur l'arbre de Merkle est introduit pour détecter les attaques de modification de paquet. Avec sa structure arborescente, il fournit une vérification efficace et sécurisée pour s'assurer qu'un paquet, qui fait partie d'un message plus grand, est reçu sans modification. Deuxièmement, le modèle de confiance proposé ajoute une surveillance en amont pour améliorer les résultats de la surveillance en aval. La surveillance en amont est effec-

tuée par le voisin à deux sauts, qui évalue la fiabilité du nœud en amont en suivant le nombre de paquets correctement authentifiés et le nombre de paquets reçus pour l'authentification. Troisièmement, le RSU enregistre les statistiques de mobilité des véhicules afin d'obtenir un poids d'actualisation pour leurs recommandations pour d'améliorer l'exactitude de l'évaluation de la confiance. Ce chapitre décrit ces techniques en détail et est organisé en six parties.

La partie 1 présente le concept d'arborescence de Merkle et décrit le processus de génération d'une valeur d'engagement Merkle pour authentifier les paquets de données entrants. En substance, un arbre de Merkle est un arbre binaire construit sur une fonction de hachage unidirectionnelle $h(\cdot)$. L'idée principale est que le nœud source fragmente un message de longueur variable en fragments de données plus petits. Chaque fragment de données est ensuite chiffré et haché à l'aide d'une fonction de hachage unidirectionnelle pour former le nœud feuille de l'arborescence Merkle. A partir des nœuds feuilles, les nœuds internes de l'arbre Merkle sont construits en hachant la concaténation des nœuds feuilles gauche et droite correspondants. Le hachage se poursuit jusqu'à ce que la valeur d'engagement soit obtenue à la racine de l'arbre. Le nœud source utilise ensuite le processus de découverte d'itinéraire, tel que le protocole de routage AODV, pour diffuser la valeur d'engagement de manière sécurisée aux autres nœuds du réseau. L'utilisation de la valeur d'engagement consiste à faciliter la vérification des paquets de données suivants reçus. Pour sécuriser la diffusion de la valeur d'engagement, le message *Route REQuest* (RREQ) contient une signature numérique pour assurer l'authenticité et l'intégrité du contenu du message. Un certificat numérique est également joint au message RREQ pour prouver la validité de la clé publique du nœud source. Si un itinéraire valide vers la destination est disponible, le nœud intermédiaire renvoie un message *Route REPly* (RREP) vers le véhicule source et stocke la valeur d'engagement dans la mémoire pour une utilisation ultérieure. La valeur d'engagement est générée et partagée publiquement uniquement si la source a un nouveau message pour la destination.

La prochaine exigence est de créer un profil de mobilité pour chaque véhicule. Il est généré en suivant les schémas de conduite du véhicule et en conservant l'état des trajets empruntés par le véhicule. Sur la base du profil de mobilité, un poids est calculé pour indiquer la fréquence à laquelle ces chemins sont empruntés et il

est utilisé pour actualiser les approbations de recommandation dans l'évaluation de confiance. En règle générale, la recommandation de confiance devrait avoir plus de poids si elle provient de véhicules qui utilisent fréquemment la route. De cette manière, l'effet des mauvaises recommandations est atténué, produisant ainsi des évaluations de confiance plus précises. Pour créer les profils de mobilité, les RSU capturent les messages balises périodiques contenant les informations de position et d'heure envoyées par les véhicules. Les RSU échangent ensuite les informations entre elles pour construire les chemins empruntés par le véhicule pendant un intervalle de temps prédéfini. Avec les informations de chemin, les profils de mobilité sont rassemblés sur une semaine où les mêmes chemins sont agrégés de manière cumulative. Par le même chemin, on entend une route qui contient le même ensemble d'identificateurs d'unités RSU. Après cela, un poids relatif à la fréquence de chaque chemin est calculé et stocké par l'unité RSU.

La partie 2 décrit le modèle MeTRO et ses sous-composants en détails. Pour faciliter l'explication, le nœud cible qui nécessite une évaluation de confiance doit être noté n_i, le précurseur de n_i est désigné par n_{i-1} et le successeur de n_i est n_{i+1}. Le premier module dans le modèle de confiance MeTRO est le module de surveillance en aval qui est exécuté par n_{i-1} pour entendre par-dessus la transmission de paquets du prochain bond, c'est-à-dire n_i. Le second module est le module de surveillance amont et est exécuté par le nœud n_{i-1}. La fonction du module amont est de vérifier l'intégrité des données reçues envoyées par le nœud n_i. La vérification du paquet de données est effectuée à l'aide du mécanisme d'authentification de l'arbre basé sur les arbres de Merkle. Pour que la vérification fonctionne, le nœud source doit ajouter le chemin d'authentification Merkle avec le bloc de données à authentifier. Le chemin authentifié contient une liste de valeurs de hachage des nœuds frères dont n_{i+1} a besoin pour authentifier le paquet de données. En utilisant la valeur d'engagement qui a été diffusée pendant le processus de découverte d'itinéraire et les valeurs de hachage fournies dans le chemin d'authentification, le nœud n_{i+1} effectue une série de hachage pour calculer le hachage racine de l'arbre Merkle. Si la valeur racine calculée est la même que la valeur d'engagement Merkle stockée par n_{i+1}, le paquet de données reçu est considéré comme valide. Si la vérification échoue, cela signifie que n_i a modifié les données ou que le message est réémis. Un compteur de paquets

est implémenté par le module amont pour suivre le nombre de paquets authentifiés avec succès. Un autre compteur de paquets est introduit pour compter le nombre de paquets reçus pour l'authentification. En utilisant ces deux valeurs, n_{i+1} calcule une métrique amont pour n_i et l'envoie à n_{i-1} via le mode diffusion. Pour éviter les tempêtes de diffusion, le TTL du paquet de diffusion est défini sur 5 sauts.

Le composant suivant est le module d'évaluation de la confiance directe qui évalue la valeur de confiance d'un nœud sur la base de la métrique aval dérivée localement et de la métrique amont de n_{i+1}. La confiance directe est formulée en tant que rapport entre le nombre d'authentifications réussies et le nombre total de paquets envoyés à n_i pour la transmission, pondérée par la proximité entre la métrique en aval et la métrique en amont. Pour déterminer le poids, *Jensen Shannon Divergence* (JSD) est utilisé. Son résultat est toujours non négatif et est borné par 1 lorsque la base logarithmique 2 est supposée. Si le résultat est proche de 0, cela implique que la métrique aval et la métrique amont sont similaires. Si le résultat est proche de 1, cela signifie que la métrique en aval et la métrique en amont sont en désaccord total. La signification du résultat est que le poids devrait être grand pour une petite valeur de JSD et faible sinon. En d'autres termes, la confiance directe d'un nœud est fonction de la manière dont les métriques aval et amont se supportent mutuellement. Le modèle de confiance MeTRO rassemble également les recommandations d'autres nœuds pour améliorer l'évaluation de la confiance. Pour répondre à ce besoin, le module de recommandation est incorporé dans le modèle de confiance MeTRO qui est responsable de l'envoi de demandes de commentaires sur n_i. Pour prendre en compte la fiabilité des recommandation reçues, n_{i-1} pondère les valeurs en fonction du profil de mobilité du recommandeur afin de calculer la valeur de confiance indirecte. Le facteur de pondération permet d'éviter toute déclaration erronée lancés par le recommandeur malveillant. Il peut être obtenu en envoyant un message de demande séparé à un RSU proche. A la réception du message de demande, la RSU vérifie le profil de mobilité de la semaine écoulée du recommandeur et renvoie au demandeur, c'est-à-dire n_{i-1}, un poids qui correspond au trajet actuel pris par le véhicule. Ce faisant, les recommandeurs qui ont pris la voie fréquemment auraient leurs recommandations de confiance affectées par un poids plus lourd.

Une fois les approbations directes et indirectes obtenues, le module de fusion les combine pour former une valeur de confiance globale à propos de n_i. La fusion utilise

le concept de la DST et est similaire à l'approche décrite au chapitre 3. Premièrement, le comportement du nœud est classé en 4 états : confiance T, non-confiance $\bar{T}$, incertitude $(T, \bar{T})$ et l'ensemble vide ϕ. Le *bpa* est ensuite affecté à chaque état de la pyramide en fonction des règles de classification et de la relation de dissymétrie entre les valeurs de confiance directe et indirecte décrites au chapitre 3. La motivation pour utiliser la DST est d'ajouter une autre dimension d'incertitude dans la quantification de confiance. Au lieu de rejeter les approbations indirectes qui s'écartent trop des observations locales selon les méthodes d'agrégation de confiance existantes, la DST les classe comme incertitude pour atténuer les effets préjudiciables des fausses recommandations. Le dernier composant du modèle de confiance MeTRO est le module de décision. Il vérifie la valeur de confiance agrégée de n_i par rapport au seuil de détection de 0.5. Si le niveau de confiance est inférieur au seuil, MeTRO blacklistera le nœud, c'est-à-dire n_i, sur la base d'un algorithme de *backoff*. Dans cet algorithme, lorsqu'un nœud est détecté comme se conduisant mal pour la première fois, il est placé sur une liste noire pendant une période prédéfinie. Une fois la temporisation expirée, le nœud défectueux est supprimé de la liste noire et rejoint de nouveau le réseau, où il peut à nouveau participer à la communication. Cependant, si le mauvais comportement est détecté pour la deuxième fois, la minuterie de recul augmentera exponentiellement par rapport au nombre de détection de mauvaise conduite. L'algorithme de *backoff* vise à pénaliser et à décourager les nœuds de se comporter de nouveau mal. En dehors de la liste noire du nœud mal géré, le module de décision effectue également une diffusion sur liste noire dans tout le réseau pour informer les autres nœuds de l'isoler de toutes communications futures. Dans le même temps, un message RERR est renvoyé au nœud source pour lancer une nouvelle découverte d'itinéraire afin de trouver un chemin exempt de nœuds mal gérés.

La partie 3 fournit une analyse de sécurité informelle du modèle de confiance MeTRO en ce qui concerne les attaques d'écoute électronique, de relecture de message, de modification de paquet et de non-répudiation. Premièrement, le modèle de confiance MeTRO est robuste contre les indiscrétions, car les fragments de données sont cryptés pour en assurer la confidentialité en utilisant des clés par paires qui sont uniques et stockées uniquement à la source et aux véhicules de destination. De plus, les clés par paire dérivées sont conservées dans le module de sécurité

matérielle (HSM) pour une protection supplémentaire contre l'altération physique. Sans connaissance des clés par paires, il est difficile pour des véhicules du réseau de déduire le texte en clair du texte chiffré. Deuxièmement, le modèle de confiance MeTRO est sécurisé contre les attaques par répétition de message en raison de la nature éphémère des clés par paires. Supposons qu'un attaquant parvienne à capturer et rejouer un message chiffré à la destination. La valeur d'engagement calculée par le nœud intermédiaire ou n_{i+1} en utilisant la copie rejouée du message chiffré et le chemin d'authentification donné ne correspondra pas à la valeur racine stockée car la clé éphémère utilisée pour chiffrer le message relu n'est pas la même clé utilisée dans la session en cours. Troisièmement, le modèle d'approbation MeTRO reste sécurisé sous les attaques de modification de paquet car il utilise le mécanisme d'authentification d'arbre basé sur Merkle. Supposons qu'un attaquant ait obtenu l'accès à la clé par paire pour la session en cours et l'utilise pour crypter son propre message. Il est infaisable de calculer la valeur de hachage correcte puisqu'une fonction de hachage cryptographique de 256 bits est utilisée. En utilisant le paradoxe de l'anniversaire, il faudrait environ 2^{128} tentatives pour générer une collision avec une probabilité de 0.5. De même, il est difficile de modifier le chemin d'authentification pour qu'il corresponde à la valeur d'engagement. Le modèle de confiance MeTRO fournit également la propriété de répudiation. L'expéditeur est tenu de calculer et de diffuser une valeur d'engagement Merkle aux usagers de la route avant que le message proprement dit ne soit transmis. La valeur d'engagement sert à enregistrer l'existence chronologique de chaque donnée afin que des nœuds intermédiaires le long du chemin d'acheminement puissent en vérifier l'intégrité. Si la valeur d'engagement stockée correspond à la valeur d'engagement calculée, cela signifie que le message provient de l'expéditeur, ce qui prouve la propriété de non-répudiation.

La partie 4 se concentre sur l'étude des performances d'authentification du modèle de confiance MeTRO en mesurant le temps nécessaire pour générer la valeur d'engagement et le temps requis pour authentifier un paquet de données. Pour atteindre cet objectif, l'algorithme de Merkle est implémenté en C++ et simulé pour une hauteur d'arbre différente allant de 1 à 10 et pour différentes tailles de message allant de 200 octets à 512 octets. L'algorithme de hachage utilisé est SHA-2 pour fournir 128 bits de sécurité. Les résultats montrent que le temps de génération

d'une valeur d'engagement est beaucoup plus élevé que le délai d'authentification d'un bloc de données. En effet, le nœud source doit hacher les blocs de données à la base de l'arbre, suivis de $(n-1)$ calculs de hachage à l'intérieur de l'arbre. En revanche, chaque nœud intermédiaire nécessite uniquement des opérations de hachage en $log_2(n)$ pour valider un bloc de données. Les résultats montrent également une augmentation de la génération et du délai d'authentification lorsqu'un message est fragmenté en une taille de message plus grande, c'est-à-dire 512 octets contre 200 octets. Ceci est dû au hachage d'un gros bloc. Lorsque la hauteur de l'arbre augmente de 1 à 10, le temps de génération et le délai d'authentification commencent à augmenter. Ce résultat suggère que le coût de construction d'un arbre Merkle sur un grand message va augmenter de manière exponentielle, ce qui n'est pas pratique pour les applications en temps réel. Une solution de contournement consiste à fractionner un message volumineux en fragments plus petits et gérables afin de limiter le calcul. Cela signifie qu'un grand message aurait désormais plus d'une valeur d'engagement pour la vérification.

La performance du délai d'authentification est également comparée à l'algorithme *Elliptic Curve Discrete Logarithm* (ECDSA) pour démontrer l'amélioration de l'efficacité sous l'effet d'une charge de messages croissante. Tout d'abord, ECDSA est implémenté en C++ et le délai d'authentification pour la vérification d'une signature est moyenné sur 100 000 passages. Le délai d'authentification moyen de l'approche Merkle est obtenu sur la base des résultats de l'expérience précédente où la hauteur de l'arbre est 10. En utilisant ces valeurs, le délai d'authentification total est évalué numériquement pour un nombre différent de messages allant de 50 à 300. Les résultats montrent que le modèle de confiance MeTRO basé sur l'approche de Merkle est inférieur de plusieurs ordres de grandeur au retard encouru par l'algorithme ECDSA. Avec un coût d'authentification inférieur, cela signifie que le modèle d'approbation MeTRO peut prendre en charge plus de messages dans le réseau. Par conséquent, il est plus évolutif et efficace que ECDSA.

La partie 5 met en évidence les améliorations de performance du modèle de confiance MeTRO sous diverses attaques, en particulier les effets des attaques de puissance de transmission limitées, des modifications de paquets et des attaques générales de largage de paquets telles que les attaques *blackhole* et *grayhole*. Un modèle

Manhattan Grid composé de 75 nœuds mobiles et de 25 RSU statiques est simulé pour imiter une architecture V-Mesh urbaine. Les mouvements du véhicule sont générés à l'aide de l'outil Bonnmotion et les nœuds mobiles se déplacent à une vitesse moyenne de 54 km/h. Ils sont configurés pour faire une pause de 5 secondes avec une probabilité de pause de 0.5 pour modéliser les conditions de feux de circulation sur les routes. La topologie est simulée en utilisant NS-3 et les résultats du MeTRO-I sont comparés au AODV de base et à une variante du MeTRO-I sans le suivi en amont appelé modèle MeTRO-II pour mettre en évidence le gain de performance des observations en amont.

Performances sous attaque de puissance de transmission limitée. Le nombre d'attaquants de puissance de transmission augmentant de 5 à 30, le taux de livraison de paquets du modèle de confiance MeTRO-I augmente de 11% et 5% par rapport aux modèles AODV et MeTRO-II. L'amélioration est due à l'introduction de JSD pour calculer un facteur de pondération dans la formule de confiance directe. JSD sert à écarter les observations basées sur la proximité entre les observations en amont et en aval.

Performances dans le cadre d'une attaque par modification de paquet. Les performances du modèle de confiance MeTRO-I sont étudiées dans le cadre d'attaques par modification de paquet. Le taux de modification des paquets est défini comme le nombre de paquets modifiés détectés par rapport au nombre de paquets envoyés. En comparant le rapport de modification des paquets entre les trois schémas (AODV de base, MeTRO-I et MeTRO-II), la fraction des paquets modifiés dans MeTRO-I est environ 20% et 30% inférieure à celle de l'AODV de base et du MeTRO-II respectivement. En effet, chaque paquet de données reçu est authentifié à l'aide du mécanisme d'authentification d'arbre basé sur Merkle. Si un attaquant modifie les paquets de données, MeTRO-I abandonnera les paquets modifiés, puis mettra à jour le nœud précurseur n_{i-1} avec les observations amont les plus récentes. En revanche, dans le modèle MeTRO-II où la surveillance en amont n'est pas utilisée, la détection des paquets modifiés peut être ignorée. Par conséquent, les paquets modifiés sont autorisés à se propager dans tout le réseau, ce qui a contribué au rapport de modification de paquet le plus élevé.

Performances dans les attaques par chute de paquets. : Dans cette expérience, deux attaques générales de largage de paquets telles que les attaques *blackhole* et *grayhole* sont considérées et les métriques de performance telles que le taux de livraison des paquets, le délai de bout en bout et le surdébit normalisé sont évaluées. Afin d'évaluer la performance sous les attaques *blackhole*, le nombre de nœuds attaquants dans le réseau varie de 5 à 30 et ils sont configurés pour laisser tomber 100% des paquets. En termes de rapport de livraison de paquets, les deux versions du modèle de confiance MeTRO, c'est-à-dire MeTRO-I et MeTRO-II, ont des performances de livraison de paquets similaires et sont meilleures que le AODV de base. Le rapport moyen de livraison de paquets est d'environ 40% alors que celui de l'AODV de base est d'environ 20%. Le taux de livraison de paquets est meilleur en raison de la possibilité de trouver un chemin alternatif autour des nœuds égoïstes pour transmettre le reste des paquets à la destination. En termes de performances de délai de bout en bout, le MeTRO-I est le plus faible à 0.2ms sur toute la gamme des véhicules à *blackhole* simulés car il s'appuie sur les informations de surveillance en amont et en aval pour sélectionner les voisins les plus fiables pour effectuer l'expédition. Les modèles AODV de référence et MeTRO-II ont des temps de retard erratiques avec de nombreuses fluctuations car ils subissent de fréquentes interruptions de routage qui nécessitent une réinitialisation du chemin de routage. De même, le surdébit de routage normalisé pour le MeTRO-I reste systématiquement faible. Cela met en évidence l'efficacité et l'évolutivité nécessaires pour faire face à l'augmentation des véhicules attaquants.

Suivant, dans le scénario d'attaque *grayhole*, il y a 20 véhicules d'attaque *grayhole* et ils sont configurés pour laisser tomber les paquets sélectivement à des taux différents allant de 10% à 100%. On observe que les deux variantes des modèles de confiance MeTRO peuvent améliorer le taux de livraison de paquets d'environ 40% malgré les différents taux de chute de paquets. L'amélioration est de près de 20% par rapport au modèle AODV de référence sans aucun modèle de confiance employé et le temps de propagation de bout en bout et les résultats de surdébit de routage normalisés du MeTRO-I sont les plus faibles. En résumé, les simulations montrent que les deux variantes des modèles de confiance MeTRO sont tout aussi efficaces dans la détection

des attaques *blackhole* et *grayhole*. Cependant, MeTRO-I est beaucoup supérieur au MeTRO-II car il est capable de se défendre contre des attaques à puissance de transmission limitée et des attaques de modification de paquet.

La partie 6 résume les principales caractéristiques du modèle de confiance MeTRO. Ce modèle de confiance plus généralisé peut supporter de nombreux types d'attaques (écoute indiscrète, relecture de message, modification de paquet, non-répudiation) ; ce qui est une tâche difficile à résoudre dans un environnement dynamique comme le réseau V-Mesh. Des simulations NS-3 extensives ont été effectuées sur le modèle *Manhattan Grid* pour mesurer le rapport de paquets, le délai de bout en bout et les attaques de largage de paquets. Le gain de performances de la surveillance amont dans le modèle MeTRO par rapport au modèle de confiance avec surveillance en aval et simple routage AODV démontre que, outre les attaques par abandon de paquets, il peut atténuer plus efficacement les attaques de transmission et de modification de paquets. L'introduction d'un mécanisme d'authentification basé sur l'arbre de Merkle s'avère également plus efficace que l'algorithme ECDSA de facto dans l'authentification des paquets reçus, démontrant ainsi une alternative viable pour fournir une authentification rapide.

Chapitre 5

Contrairement aux deux chapitres précédents, le chapitre 5 traite du défi de se débarrasser d'une PKI et de la latence associée qui n'est pas très compatible avec les applications de sécurité des véhicules. A cette fin, un protocole de gestion de clés sécurisé et authentifié (SA-KMP, *Secure and Authenticated Key Management Protocol*) est proposé. Il combine le travail de deux mécanismes existants, à savoir PKR (*Public Key Regime*) et la distribution de clé réseau 3D. L'objectif de PKR est de faciliter la distribution de clés publiques certifiées et mises à jour, de sorte que la gestion des CLR à forte besoin en ressources puisse être éliminée. D'autre part, la distribution des clés de réseau 3D vise à établir des clés symétriques pour sécuriser les communications véhiculaires plutôt que d'utiliser une cryptographie asymétrique coûteuse. Cependant, un examen de ces deux systèmes révèle certaines vulnérabilités.

Dans la proposition PKR d'origine, les clés publiques des véhicules sont stockées dans un répertoire de fichiers public et diffusées à toutes les unités RSU du réseau V-Mesh. Le rôle de la RSU est de distribuer des clés publiques aux véhicules pour faciliter la communication. Avec cette approche, il n'est pas nécessaire de distribuer des certificats numériques et de télécharger de grandes listes CRL. Cependant, étant donné que les RSU doivent desservir de nombreuses requêtes provenant des véhicules dans une région, cette approche peut créer un goulot d'étranglement sur l'unité RSU, provoquant ainsi des attaques DoS et une compromission de la disponibilité du service. En outre, le RSU peut comploter avec un véhicule malveillant pour émettre une clé publique appartenant au véhicule de collusion.

Dans le schéma de distribution de clé réseau 3D, chaque nœud se voit attribuer un ensemble de clés pré-partagées qui est décrit par une équation plan. Pour trouver des clés communes entre deux nœuds communicants quelconques, chaque nœud doit résoudre un système d'équations linéaires où la solution représente l'emplacement de la clé pré-partagée que les deux nœuds ont en commun. Puisque les clés pré-partagées sont préchargées sur chaque nœud, un adversaire peut capturer physiquement le nœud pour compromettre l'espace des clés en entier. Le schéma de distribution des clés 3D est également vulnérable aux attaques DoS car les informations nécessaires à la résolution du système d'équations linéaires doivent être signées numériquement pour garantir l'authenticité et l'intégrité. Pour cette raison, un nœud malveillant peut envoyer de nombreux messages contrefaits pour forcer le destinataire à effectuer une vérification inutile afin d'épuiser ses ressources.

En résumé, ce chapitre se concentre sur le renforcement de la PKR et des schémas de distribution de clé réseau 3D qui composent les attaques SA-KMP, en particulier DoS et l'attaque par capture de nœud. Pour faciliter la discussion, ce chapitre est divisé en cinq parties.

La partie 1 présente les objectifs de conception et discute des moyens d'améliorer les schémas de distribution PKR et de grille-clé 3D pour atténuer les attaques DoS et l'attaque par capture de nœuds. Les objectifs de conception sont définis de deux manières : performance et sécurité. En termes de performance, le SA-KMP proposé doit être efficace, léger et s'adapter à la croissance du réseau avec une bonne évolutivité. En termes de sécurité, les messages doivent être protégés pour assurer

l'intégrité, l'authenticité, la non-répudiation, la confidentialité et la protection de la vie privée. Sur la base de ces objectifs de conception, plusieurs techniques ont été proposées pour améliorer les propriétés de la PKR et les schémas de distribution de la grille-clé 3D.

Améliorations PKR : Pour répondre aux attaques DoS, un protocole de preuve de connaissance basé sur l'algorithme de signature *Elliptic Curve-Schnorr* est construit et intégré dans l'approche PKR. L'idée principale est que la RSU génère une signature contenant une valeur d'engagement aléatoire pour le véhicule demandeur à vérifier. Le véhicule doit vérifier la signature pour extraire la valeur d'engagement et l'utilise pour calculer un hash que l'unité RSU doit vérifier. Si la preuve de hachage est correcte, la RSU procède à l'émission de la clé publique demandée au véhicule. Étant donné que le véhicule demandeur doit dépenser certaines ressources de l'unité centrale pour vérifier la signature, le protocole peut empêcher les véhicules malveillants d'initier des attaques DoS et empêcher l'écrasement de l'unité RSU. En termes de sécurité du protocole, la signature EC-Schnorr est basée sur la difficulté du calcul des logarithmes discrets et l'impossibilité d'inverser une fonction de hachage cryptographique unidirectionnelle. Par conséquent, il est impossible de calculer la valeur d'engagement ou de forger un hash valide pour vérification. En termes d'efficacité, la signature EC-Schnorr nécessite seulement une exponentielle modulaire, une addition et une multiplication pour générer la signature, donc, elle est légère.

Deuxièmement, pour alléger les attaques par collusion lorsque la RSU peut s'entendre avec un véhicule malveillant pour émettre une clé publique incorrecte, le TA crée une signature pour chaque tuple ⟨ID du véhicule, clé publique⟩ stockée dans le répertoire public de fichiers de sorte que lorsque la RSU reçoit le message de demande de clé publique, il l'ajoute également au message de réponse de clé publique au demandeur. De cette manière, le véhicule demandeur peut vérifier l'exactitude de la clé publique émise en vérifiant la signature de TA.

Troisièmement, pour faciliter une communication transparente dans un délai minimal, les clés publiques des RSU sont diffusées à tous les véhicules sous la forme d'un fichier public. Pour distinguer les deux répertoires, le véhicule est assigné à une RSU-Dossier Public (RSU-PFD) qui contient les clés publiques de toutes les RSU,

et la RSU reçoit un répertoire Vehicle Public File Directory (VPFD) qui ne contient que clés publiques des véhicules. En disposant de deux répertoires de clés publiques différents, il réduit le temps de stockage des clés et permet d'accélérer les communications V2I, permettant ainsi aux communications de commencer immédiatement sans avoir besoin d'envoyer le certificat et de télécharger les listes CRL. Comme le nombre d'unités RSU est fixe, le RSU-PFD conservé par les véhicules nécessite rarement une mise à jour. Cependant, les véhicules peuvent toujours recevoir la copie mise à jour du RSU-PFD des fabricants d'équipement d'opérateur (OEM) dans le cadre de la maintenance périodique du véhicule, si nécessaire. D'autre part, puisque les RSU sont statiques et bien connectées en utilisant la topologie maillée, le VPFD est mis à jour en temps réel.

Améliorations de la distribution de clés réseau 3D. Pour contrer la divulgation des clés due aux attaques de capture de nœuds, des fonctions de hachage sont introduites pour générer les clés dynamiquement. Pour atteindre ce but, chaque partie communicante joue un rôle dans le choix d'une valeur de nonce aléatoire pour générer les clés communes. Après cela, les deux valeurs aléatoires sont hachées ensemble à plusieurs reprises. Le nombre de fois à hacher dépendra des points de solution du groupe d'équation plan. Puisque les clés communes sont générées à la demande et sont uniques en raison des valeurs aléatoires choisies par les parties par session, la version améliorée du schéma de clé de grille 3D est protégée contre le problème e divulgation de clé par rapport à l'approche par préchargement. Même si un nœud est capturé par un adversaire, seule la communication du nœud affecté est compromise. En outre, les énormes coûts de communication associés à la mise à jour de l'espace des clés (si l'approche de préchargement est utilisée) sont évités. En plus de générer des clés à la demande, le schéma de distribution de clés 3D est également complété par le protocole de preuve de connaissance pour combattre les attaques DoS. En d'autres termes, chaque partie communicante doit se prouver mutuellement l'intention de dériver les clés par paire en engageant certaines de leurs ressources CPU pour échanger et vérifier la signature EC-Schnorr afin d'en extraire la valeur d'engagement.

La partie 2 détaille les divers composants du cadre SA-KMP pour illustrer le flux

de travail des communications sécurisées. Cette partie explique également comment les schémas de distribution de clés réseau PKR et 3D sont adaptés pour atténuer les attaques DoS et de capture de nœud.

Configuration du système : Ce composant est responsable de la génération de paramètres à l'échelle du système pour prendre en charge les opérations SA-KMP. Le TA crée deux répertoires : RSU-PFD et VPFD et stocke la paire ID/clé publique des RSU et des véhicules. En outre, le TA crée et stocke une signature pour chaque entrée dans le VPFD pour empêcher une RSU de collusion d'émettre une mauvaise clé publique. Le RSU-PFD et le VPFD sont ensuite diffusés sur chaque véhicule et RSU respectivement. Les avantages de la diffusion de répertoires à jour sont triples : (1) pas besoin de télécharger des listes CRL énormes, (2) cela réduit le temps et la complexité de la vérification des certificats et (3) cela améliore la rapidité et la fraîcheur des clés publiques. Après cela, le TA génère un ensemble d'équations plan et trois fonctions de hachage et les distribue à toutes les entités pour calculer les clés par paires.

Demande de clé publique : ce composant implémente le protocole de preuve de connaissance pour atténuer l'impact des attaques DoS. L'idée est de forcer les véhicules demandeurs à valider certaines ressources de l'unité centrale pour vérifier la signature de l'unité RSU avant l'émission de la clé publique. Le protocole se compose de six étapes.

Étape 1 et Étape 2 : Chaque nœud communicant sélectionne deux valeurs aléatoires, la première valeur aléatoire étant un scalaire multiplié par un point de base pour générer une valeur d'engagement C dans le groupe de courbes elliptiques, et la seconde valeur aléatoire R est une valeur nonce pour empêcher une attaque de rejeu. Le nonce R et la valeur d'engagement C peuvent tous deux être pré-calculés hors ligne pour une meilleure efficacité. La valeur d'engagement C sert de preuve de connaissance pour indiquer que le prouveur, c'est-à-dire le véhicule, a vérifié la signature de l'unité RSU.

Étape 3 : La communication commence avec un véhicule qui envoie un message de

requête de clé publique à la RSU. Ce message contient son propre ID, l'ID de la RSU et le nonce aléatoire R_v du véhicule (calculée à l'étape 1 et 2).

Étape 4 : Lorsque la RSU reçoit le message de requête du véhicule, elle génère une signature EC-Schnorr consistant en un défi e_R et une réponse S_R. Puis elle envoie la paire de signature au véhicule. Le défi e_R est calculé en hachant le message, la valeur de nonce aléatoire du véhicule R_v et la RSU R_R (calculée aux étapes 1 et 2) en incluant la coordonnée en x de la valeur d'engagement de RSU $x_{C_\mathcal{R}}$ (calculée à l'étape 1 et 2).

Étape 5 : Lorsque le véhicule reçoit la signature de la RSU, il récupère la clé publique k_R^+ de la RSU du RSU-PFD pour recalculer la valeur d'engagement de la RSU, $\overline{C_\mathcal{R}}$, en utilisant les paramètres e_R, S_R et k_R^+. Ensuite, le véhicule calcule sa propre version du défi de le RSU $\overline{e_R}$ en hachant le message concaténé avec les valeurs aléatoires du véhicule R_v et de la RSU R_R et l'abscisse de la valeur d'engagement auto-évaluée $\overline{x_{C_\mathcal{R}}}$. Si le défi calculé $\overline{e_\mathcal{R}}$ correspond à celui envoyé par l'unité RSU, cela implique que l'unité RSU est légitime. Le véhicule génère alors sa propre signature EC-Schnorr et une preuve notée v_v et envoie le tuple (v_v, e_v, S_v) à la RSU pour compléter le processus d'authentification. La preuve v_v est une valeur de hachage et contient la coordonnée x de la valeur d'engagement auto-évaluée $\overline{x_{C_\mathcal{R}}}$. Il est utilisé par le RSU pour vérifier que le véhicule a vérifié la signature.

Étape 6 : Avant que le RSU vérifie la paire de signature (v_v, e_v, S_v) du véhicule. Il vérifie d'abord que la preuve de hachage v_v fournie par le véhicule est correcte en hachant les valeurs nonce aléatoires (R_R, R_v) et l'abscisse de sa propre valeur d'engagement x_{C_R} générée à l'étape 2. La paire de signature du véhicule (e_v, S_v) n'a besoin d'être vérifiée que si la preuve fournie v_v correspond. Si la preuve est incorrecte, cela implique que le véhicule n'a pas vérifié la signature à l'étape 5. Après vérification de la preuve, la RSU continue à localiser la clé publique k_v^+ du véhicule dans le VPFD et effectue le processus de vérification pour calculer la valeur d'engagement du véhicule, c'est-à-dire $\overline{C_\mathcal{V}}$ en utilisant e_v, S_v et k_v^+. Similaire au processus de vérification à l'étape 5, le RSU vérifie si la version calculée du défi $\overline{e_v}$ correspond à celle envoyée par le véhicule. Si le défi correspond, la RSU répond au véhicule par

un message contenant l'ID demandé, la clé publique demandée et la signature du TA sur la liaison demandée ⟨ID, clé publique⟩ récupérée du VPFD. Le message entier est crypté en utilisant la clé publique du véhicule k_v^+ pour assurer la confidentialité. L'inclusion de la signature du TA vise à assurer l'intégrité et l'authenticité du message. Lorsque le véhicule reçoit le message de réponse, il déchiffre le message en utilisant sa propre clé privée et vérifie la signature du TA à l'aide de la clé publique de TA. Si la signature est correcte, cela signifie que la clé publique demandée émise par l'unité RSU est correcte.

Accord de clé : Ce composant est responsable de l'échange de matériaux de chiffrement pour établir des clés par paires et des clés de groupe. Il protège de DoS, de sorte que le même protocole de preuve de connaissance décrit dans le composant de demande de clé publique est incorporé. La seule différence est que les messages échangés contiennent des informations supplémentaires telles que le nonce généré aléatoirement et les informations de localisation GPS de l'unité RSU et du véhicule. Le protocole d'accord de clé par paire peut également être appliqué aux communications V2V. Dans ce cas, le véhicule demandeur doit d'abord contacter la RSU pour demander la clé publique du véhicule cible en exécutant d'abord le composant de demande de clé publique. Par la suite, le reste de la procédure dans V2V suit les mêmes étapes dans la communication V2I. Dans l'accord de clé de groupe, l'échange des clés reste entre le véhicule du cluster et la RSU. Mais le processus est légèrement différent dans la mesure où la RSU doit aider le leader du cluster à authentifier les véhicules du groupe.

Dérivation de clé : Ce composant implémente le schéma de distribution de clés 3D pour dériver des clés par paires et des clés de groupe pour la communication. En utilisant les informations échangées telles que les emplacements GPS et les valeurs des nonces aléatoires collectées à partir du composant d'accord de clé, chaque nœud résout un système de 3 équations plan pour déterminer les points de solution. Plus précisément, le véhicule introduit sa propre localisation GPS (i_v, j_v, k_v) dans la première équation du plan et substitue l'emplacement GPS (i_R, j_R, k_R) de l'unité RSU dans la seconde équation. Dans l'équation du troisième plan, le véhicule substitue sa propre valeur de nonce, la valeur de nonce du RSU et le hachage des deux valeurs

de nonce. L'utilisation des valeurs aléatoires aléatoires dans la dernière équation du plan garantit l'unicité des clés dérivées. Cela garantit qu'un autre véhicule qui communique avec la même unité RSU ne recevra pas les mêmes clés. Comme il y a 3 ensembles d'entrées à substituer dans N équations de plan, il y a $N(N-1)(N-2)$ groupes d'équations à résoudre qui produisent $N(N-1)(N-2)$ solutions uniques. L'ensemble des solutions est représenté par $S = \{(x,y,z)_1, (x,y,z)_2, \cdots, (x,y,z)_d\}$ où (x,y,z) désigne une solution point et d indiquent le nombre de solutions dans l'ensemble. Sur la base du point de solution, les clés par paires sont obtenues en hachant la concaténation des deux valeurs aléatoires et l'ID du demandeur à plusieurs reprises. Le nombre d'opérations de hachage est déterminé par la valeur de x, y et z donnée par le point de solution (x,y,z). De même, l'unité RSU effectue les mêmes étapes pour générer les clés.

Une fois que les clés par paires sont disponibles, le véhicule peut utiliser n'importe quelle clé dans le pool de clés $N(N-1)(N-2)$ pour crypter et déchiffrer les messages. Un HMAC est ajouté avec le message crypté pour prouver l'authenticité et l'intégrité. Le message contient également un horodatage pour empêcher les attaques de rejeu. Lorsque la RSU reçoit le message du véhicule, elle utilise la même clé indexée par un identifiant de clé pour déchiffrer le message. Cela garantit que seuls les destinataires légitimes qui ont participé à l'algorithme d'accord de clé et qui ont échangé des matériaux de clé peuvent convenir des mêmes clés. Plusieurs clés peuvent également être combinées à l'aide d'un opérateur XOR pour former une clé composite. Cela ajoute de la complexité à la clé et augmente le niveau de sécurité dans la communication.

Dans l'accord de clé de groupe, après que les clés de groupe ont été établies, la RSU aide à disséminer les clés de tous les véhicules authentifiés dans le groupe où elles sont encapsulées dans un message crypté avec une clé de cryptage aléatoire choisie dans le pool de clés. Pour que le véhicule reçoive l'ensemble des clés de groupe, la RSU doit transmettre la clé de cryptage aux véhicules du groupe. Pour ce faire, la RSU crypte la clé de cryptage à l'aide de la clé publique du véhicule et ajoute une signature pour prouver l'authenticité et l'intégrité de la clé de cryptage. Après que le véhicule a vérifié la clé de chiffrement, il déchiffre le message pour récupérer les clés de groupe pour la communication. Par la suite, le leader du cluster peut utiliser n'importe quelle clé de groupe dans le pool de clés pour chiffrer tous les

messages de manière symétrique dans le groupe. Un HMAC est ajouté pour prouver l'authenticité et l'intégrité du message envoyé. Seuls les véhicules avec la clé de groupe correcte peuvent déchiffrer le message et les clés de groupe sont indexées de sorte que les autres véhicules du groupe sachent quelle clé est utilisée pour déchiffrer le message.

Révocation de clé : Ce composant est responsable de la révocation des clés et se compose de deux parties. La première partie implique la révocation des clés publiques. Dans le composant de demande de clé publique et dans le composant d'accord de clé, si un véhicule échoue au protocole de preuve de connaissance, l'unité RSU envoie un message au TA pour révoquer la clé publique du véhicule. Le TA mettra à jour le VPFD et le diffusera en temps réel à tous les RSU en utilisant un réseau filaire sécurisé. Cela signifie que les communications futures avec le véhicule révoqué seront interdites à moins qu'il ne s'enregistre de nouveau auprès du TA pour recevoir un autre ensemble de paires de clés publiques/privées.

La deuxième partie se concentre sur l'actualisation des clés par paires ou des touches de groupe. En supposant que chaque véhicule et chaque RSU envoient des messages entre 100 ms et 300 ms selon la norme WAVE et en supposant que la clé partagée ne peut être utilisée qu'une seule fois par message, la limite inférieure et la limite supérieure du temps d'expiration peuvent être déterminées comme $0.1 * N(N-1)(N-2)$ et $0.3 * N(N-1)(N-2)$ respectivement où l'expression $N(N-1)(N-2)$ indique le nombre de clés possibles et N désigne le nombre d'équations plan. Cela signifie que le véhicule n'a pas besoin d'établir des clés partagées chaque fois qu'il passe une RSU, ce qui peut aider à réduire considérablement les frais généraux de regénération de clés. Il est également possible de déterminer l'heure d'expiration d'une clé en fonction du nombre d'utilisations. Un compteur peut être implémenté pour suivre le nombre de messages chiffrés à l'aide d'une clé spécifiée. Une fois que la clé dépasse la limite prédéfinie par le demandeur de clé, une autre clé est sélectionnée dans la liste. Lorsque toutes les clés sont épuisées, les nœuds correspondants doivent à nouveau réinitialiser la procédure d'accord de clé.

La partie 3 fournit une analyse de sécurité informelle du cadre SA-KMP dans le cadre d'attaques classiques telles que DoS, écoute électronique, rejeu, usurpation

d'identité de nœud et collusion.

Attaques DoS. Les attaques DoS sont atténuées par le protocole de preuve de connaissance. Dans la demande de clé publique et le composant d'accord de clé, l'unité RSU doit vérifier si le véhicule a fourni la bonne valeur de hachage. Cela signifie que le véhicule doit dépenser des ressources CPU pour vérifier la signature de la RSU afin d'extraire la valeur d'engagement correcte pour le hachage. Par conséquent, les attaquants malveillants ne bénéficieront de rien s'ils lancent des attaques DoS. Le véhicule malveillant peut également tenter de contourner la vérification de la signature de la RSU en forgeant une valeur de hachage valide pour rompre le protocole. Cependant, cela n'est pas possible en raison de la propriété de la fonction de hachage unidirectionnelle. Le délai d'authentification de l'utilisation du protocole preuve de connaissance dans le framework SA-KMP est également analysé en considérant un scénario d'attaque DoS où un attaquant initie un nombre de messages non valides égal à 10%, 20% et 30% du nombre de messages valides. Le délai d'authentification est défini comme le temps nécessaire à l'unité RSU pour traiter les demandes des véhicules. La performance simulée est ensuite comparée au schéma PKR et EMAP. Les résultats montrent que le SA-KMP est supérieur à l'approche PKR du fait que la RSU doit seulement effectuer une étape légère pour valider l'épreuve de hachage contenant sa valeur d'engagement. Si la preuve est incorrecte, l'unité RSU ignorera la vérification de la signature. D'un autre côté, dans l'approche PKR où seules les signatures sont ajoutées au message, les RSU doivent vérifier chaque signature. Le cadre SA-KMP fonctionne également mieux que le schéma EMAP qui utilise l'ECDSA car le protocole de preuve de connaissance utilise une signature EC-Schnorr plus efficace.

Attaques à l'aveuglette. SA-KMP est sécurisé car tous les messages sont cryptés pour en assurer la confidentialité. Dans le composant de dérivation de clé, deux nœuds communicants ont en commun $N(N-1)(N-2)$ clés distinctes où N désigne le nombre d'équations. Cela implique que les chances de deviner la clé dérivée sont $1/(N(N-1)(N-2))$. Ce qui est très faible si N est grand. De plus, si le cryptage SHA-1 est utilisé, la complexité de calcul pour trouver la clé correcte est $O(2^L)$ où $L =$ la longueur de la clé en bits. En outre, la difficulté de craquer les clés peut être renforcée en utilisant une clé de session composite créée par XORing certaines des

clés dérivées.

Attaques par effraction. SA-KMP est sécurisé car tous les messages sont des attaques de relecture : des valeurs aléatoires sont utilisées dans le composant de demande de clé publique et le composant d'accord de clé. Le nonce est long d'au moins 128 bits, ce qui signifie que la probabilité d'obtenir le même nonce aléatoire pour deux sessions de communication différentes est égale à $1/2^{128}$. En outre, tous les messages sont horodatés pour assurer la fraîcheur. Si le message n'est pas reçu dans une période tolérable, la vérification HMAC échouera et le paquet sera ignoré.

Attaque d'usurpation d'identité. Un véhicule malveillant peut lancer une attaque d'usurpation d'identité pour tromper d'autres véhicules ou la RSU qu'il est le véritable initiateur du message. Le véhicule malveillant peut également forger de fausses cartes d'identité pour échapper à la détection lors d'accidents ou de crimes. Cela n'est pas possible dans le cadre de SA-KMP parce que les véhicules et les RSU utilisent des identifiants de nœud et des clés publiques à jour et déjà certifiés disponibles dans le RSU-PFD et le VPFD pour s'authentifier mutuellement dans le composant d'accord de clé. L'utilisation de clés partagées pour le chiffrement symétrique fournit également un certain niveau d'authentification car les clés partagées ne sont connues que d'un groupe de véhicules. Par conséquent, chaque nœud peut être sûr qu'il communique avec la bonne partie et non avec certains imitateurs.

Attaques par collusion. Deux cas sont considérés. Premièrement, l'unité RSU peut s'associer à un autre véhicule malveillant pour émettre une mauvaise clé publique dans le composant de demande de clé publique. Deuxièmement, les nœuds communicants peuvent se regrouper et regrouper les clés préchargées pour exposer l'espace des clés. Dans le premier cas, chaque entrée stockée dans le VPFD contient une signature signée par le TA. La RSU doit joindre la signature du TA avec la clé publique demandée au véhicule pour vérification. Par conséquent, il n'y a aucun moyen pour la RSU de collusion avec un autre véhicule pour manipuler et émettre une réponse de clé publique signée valide. Dans ce dernier cas, le composant de dérivation de clé est conçu pour générer des clés de paires et de groupe de manière dynamique sur la base des entrées actuelles des équations plan au lieu de la méthode

de préchargement. Cette approche augmente la résilience aux attaques de collusion ainsi qu'aux attaques de capture de nœuds.

Vérification formelle : Les propriétés de sécurité du framework SA-KMP sont également analysées formellement à l'aide d'un vérificateur de protocole cryptographique automatique appelé ProVerif. Il est utilisé pour modéliser le composant d'accord de clé et le composant de dérivation de clé. Les résultats de vérification montrent que SA-KMP peut satisfaire à la propriété de secret, c'est-à-dire qu'un adversaire ne peut pas obtenir les clés par paires dérivées. Le SA-KMP est également authentifié par le protocole de preuve de connaissance et les répertoires de fichiers publics préchargés diffusés aux RSU et aux véhicules.

La partie 4 analyse les performances de SA-KMP en ce qui concerne (1) les surdébits de transmission, (2) le temps système de stockage et (3) la latence. Enfin, (4) la complexité de calcul de la composante de dérivation de clés en termes de temps de génération de clés est étudiée pour illustrer sa haute efficacité. Dans la première partie, la performance est comparée à l'approche PKI et PKR basée sur les certificats et dans la seconde partie, le temps de génération de la clé 3D est comparé au DH/ECDH en utilisant une longueur de clé différente.

Dépassement de transmission. Le surdébit est évalué en considérant un scénario dans lequel un véhicule envoie un paquet à un RSU ou vice versa. Les résultats montrent que l'approche PKI basée sur un certificat entraîne la surcharge de transmission car le certificat doit être joint au message. Par l'approche SA-KMP et PKR, il n'est pas nécessaire d'attacher le certificat car le TA et le véhicule reçoivent chacun une copie vérifiée du répertoire de clé publique. L'impact de l'augmentation de la charge de messages de 1 à 30000 sur le surdébit de transmission est davantage illustré. On peut voir que le temps de transmission de SA-KMP est inférieur de 30% à l'approche PKI basée sur un certificat, ce qui signifie que 69% de la bande passante de communication est sauvegardée sans certificat de transmission.

Frais généraux de stockage. La taille du VPFD et du RSU-PFD détenus par chaque RSU et chaque véhicule est estimée en considérant les paramètres suivants : clé pu-

blique du véhicule = 29 octets, clé publique du RSU = 33 octets, taille de la signature TA = 64 octets et véhicule et l'identité de RSU = 4 octets. Les résultats de la comparaison montrent que la RSU dans le cadre SA-KMP a une exigence de stockage plus élevée qu'avec le schéma PKR en raison de la signature TA supplémentaire qui est stockée dans le VPFD. Chaque véhicule dans le cadre SA-KMP doit également stocker le RSU-PFD qui est une exigence de stockage supplémentaire. Cependant, ce stockage supplémentaire est nécessaire pour augmenter l'efficacité afin que les communications V2I puissent commencer sans avoir à échanger la clé publique de l'unité RSU qui contribue à la latence élevée. Dans l'approche PKI basée sur un certificat, le temps de stockage peut varier. Dans certains cas, il peut être inférieur au cadre SA-KMP en fonction du nombre de certificats à stocker. Cependant, l'inconvénient de l'approche PKI est que la latence est élevée en raison de la diffusion des CRL et de la vérification des certificats.

Latence. La performance est analysée en mesurant le temps nécessaire à un véhicule pour établir une communication V2V avec un autre véhicule. Dans le cadre SA-KMP, la latence consiste à exécuter le composant de demande de clé publique pour extraire la clé publique du nœud de destination à partir de l'unité RSU. Dans l'approche PKR, la latence fait référence à la même procédure de demande de clé publique mais sans le protocole de preuve de connaissance. Dans l'approche PKI basée sur les certificats, la latence consiste à télécharger la liste CRL et à vérifier la validité du certificat. Les résultats montrent que le cadre SA-KMP a une latence de communication V2V plus faible que l'approche PKR même si le protocole de preuve de connaissance est introduit. Le cadre SA-KMP peut également prendre en charge la communication V2I avec une faible latence de réseau qui n'a pas été prise en compte par l'approche PKR. La latence de communication du cadre SA-KMP est également inférieure de plusieurs ordres de grandeur à l'approche PKI basée sur les certificats. Le cadre SA-KMP est supérieur car 1) aucun certificat n'est envoyé dans la transmission du message et 2) l'opération de signature EC-Schnorr est plus rapide que l'ECDSA car elle n'implique pas d'inversions modulaires.

Temps de génération de la clé. Les effets de la modification de la taille du réseau et du nombre d'équations plan sur le temps de génération de la clé sont analysés et

comparés aux approches ECDH et DH. Le temps de génération de clé est mesuré en implémentant la composante de dérivation de clé du cadre SA-KMP dans un programme C. Dans la première expérience, le nombre d'équations plan est fixe et la taille du réseau est variée. Les résultats de la simulation montrent que le schéma SA-KMP a un temps de génération de clé beaucoup plus faible par rapport au protocole DH dans les différentes tailles de réseau. Le cadre SA-KMP surpasse également les protocoles ECDH-256 et ECDH-224 lorsque la taille du réseau est inférieure à 9 km et 8 km respectivement. Dans la deuxième expérience, la taille du réseau est maintenue à 5 km et le nombre d'équations du plan varie de 3 à 10. Les résultats montrent que le temps de génération de la clé augmente exponentiellement avec le nombre d'équations plan rendant le SA-KMP inefficace. Pour résoudre ce problème, les deux nœuds communicants peuvent choisir n'importe quel sous-ensemble des N équations du plan à résoudre au lieu de toutes les N équations. Cela raccourcira le temps de génération de la clé et rendra le framework SA-KMP hautement configurable et flexible.

Complexité de calcul. La complexité de l'établissement de clés par paires est évaluée dans les cycles de CPU pour deux cas : communications V2I et V2V. Dans la communication V2V, la complexité consiste à exécuter la requête de clé publique et les composants d'accord de clé. Dans la communication V2I, la complexité est dominée par le composant de dérivation de clé. Les résultats de la simulation montrent que les communications V2V nécessitent 70.84 mégacycles, soit 57.4% de plus que les communications V2I (29.30 mégacycles) en raison des opérations de demande de clé publique supplémentaires. Le cycle CPU total pour chaque scénario de communication peut également être converti en temps de calcul réel en fonction des spécifications matérielles. Supposons que l'OBU dans un véhicule est de 500 MHz, le temps de calcul pour les scénarios de communication V2I et V2V sera de 58.60 ms et 141.68 ms respectivement, ce qui suggère que le cadre SA-KMP est réalisable dans les paramètres du monde réel.

La partie 5 conclut ce chapitre en résumant les principales caractéristiques du cadre SA-KMP. Fondamentalement, il tire parti de l'approche PKR et du schéma de distribution de clé réseau 3D pour réduire la latence et la complexité de l'uti-

lisation d'une PKI basée sur des certificats. Grâce à l'analyse numérique, il est prouvé que le système SA-KMP est compatible avec les contraintes de sécurité véhiculaire contrairement au schéma PKI basé sur les certificats. Deuxièmement, le framework SA-KMP est évolutif sans liste de révocation. Troisièmement, les demandes de transmission sont inférieurse. Bien que la signature du TA doive être stockée dans le VPFD, ce qui nécessite un stockage supplémentaire, les besoins de stockage sont encore beaucoup plus faibles que la capacité de stockage d'un OBU moderne avec un espace de stockage de 8 Go. Plusieurs programmes C ont également été développés pour simuler les propriétés de sécurité du cadre SA-KMP proposé et un ProVerif formel a été développé pour valider les primitives cryptographiques de l'accord de clé et des composants de dérivation de clé. Les résultats montrent que le framework SA-KMP peut non seulement résister aux attaques classiques (écoute électronique, modification de données, rejeu, répudiation, collusion ...), mais il est également conçu spécifiquement pour dissuader les attaques par déni de service.

Chapitre 6

En conclusion, le but principal de cette thèse était de promouvoir une communication fiable et sécurisée dans un réseau V-Mesh. À cette fin, trois modèles de sécurité ont été proposés qui s'inspirent de deux aspects : le modèle de confiance et le système de gestion des clés. La première partie du chapitre résume les principaux résultats et la deuxième partie traite des travaux futurs.

La partie 1 rappelle le modèle de confiance DS-Trust. Ce modèle de confiance tire des déductions à partir de relations humaines dans un environnement social pour gérer les recommandations de confiance. Sur la base de cette idée, les principes DST sont adoptés pour modéliser l'écart entre les recommandations de confiance reçues et les recommandations de confiance déjà reçues et classées comme incertaine. Par la suite, le poids des recommandations de confiance est réajusté en conséquence avant qu'elles ne soient combinées avec le dossier de confiance directe pour former la confiance finale. Les résultats de l'analyse numérique et de la simulation démontrent que le modèle DS-Trust peut atténuer les attaques de *badmouthing* et bourrage d'urnes et améliorer le débit du réseau en évitant les nœuds égoïstes pendant le

routage. L'utilisation du raisonnement d'incertitude dans DST rend également le modèle plus robuste contre un plus grand nombre de mauvais recommandeurs par rapport aux approches de confiance existantes.

Le second modèle de confiance s'appelle MeTRO et utilise deux techniques. La première technique consiste à permettre à un nœud d'exploiter les rapports de surveillance en amont d'un nœud à deux bonds pour renforcer ses propres observations de surveillance en aval. La deuxième technique introduit un mécanisme d'authentification appelé l'authentification basée sur les arbres de Merkle pour lutter contre les modifications de paquets. Les résultats de la simulation indiquent que le modèle de confiance MeTRO est capable de détecter les attaques de puissance de transmission limitées, les attaques de modification de paquet incluant les attaques *blackhole* et *grayhole*. Les résultats indiquent également que l'utilisation de l'authentification basée sur les arbres de Merkle est beaucoup plus efficace que l'ECDSA en termes de délai d'authentification. Au mieux de nos connaissances, MeTRO est le premier travail qui intègre un mécanisme d'authentification pratique dans un protocole de routage pour détecter les modifications de paquets. Deuxièmement, MeTRO est le premier qui évalue la fiabilité d'un nœud en considérant à la fois le comportement de transfert d'un nœud et l'intégrité des paquets transférés.

La deuxième partie de la thèse a porté sur les problèmes d'utilisation de la PKI et de la cryptographie asymétrique pour sécuriser le réseau V-Mesh, c'est-à-dire une latence élevée et une faible extensibilité. En réponse, un cadre de sécurité appelé SA-KMP efficace, évolutif et sécurisé est proposé. Il est réalisé au moyen de la diffusion de référentiels contenant les clés publiques d'autres utilisateurs de chaque réseau et en utilisant la distribution de clé 3D pour établir un groupe de clés symétriques pour la communication pour décharger les opérations asymétriques intensives de calcul. Un protocole de preuve de connaissance est également incorporé pour prévenir les attaques DoS. Dans le mécanisme de distribution de clé réseau 3D, les clés pré-partagées ne sont pas pré-chargées mais générées à la volée pour empêcher les attaques de capture de nœud. Une validation de sécurité formelle et informelle a été effectuée pour démontrer que SA-KMP est sécurisé contre un large éventail d'attaques et que le protocole d'accord de clé satisfait les propriétés d'authenticité et de secret. Les résultats de la simulation indiquent que le SA-KMP a un temps de transmission plus faible et une latence plus faible par rapport à la PKI basée sur

le certificat. Bien que le temps de génération de clé dans le protocole d'accord de clé augmente avec la taille du réseau et le nombre d'équations de plan utilisées, des compromis pourraient être faits pour équilibrer les besoins de sécurité et les coûts de complexité.

La partie 2 traite d'autres travaux liés aux trois modèles proposés et est divisée en deux sections. La première section met en évidence les limites des modèles de confiance proposés et propose plusieurs extensions pour les surmonter. La deuxième section présente des améliorations au cadre SA-KMP.

Dans les modèles de confiance DS-Trust et MeTRO, un seuil de détection de confiance fixe est supposé qui ne s'adapte pas bien à l'environnement sans fil non fiable qui cause la chute de paquets. En tant que travail futur, il est possible d'estimer la perte de paquets en sans fil pour définir le seuil de confiance de manière adaptative. Cela permettrait au modèle de confiance de mieux différencier les pertes de paquets malveillants des pertes de paquets. Cette amélioration permettrait aux deux modèles de confiance de détecter l'attaquant *greyhole* avec une plus grande précision et d'améliorer sensiblement le PDR et le débit. Deuxièmement, l'évaluation des performances actuelles des deux modèles de confiance est basée sur le protocole de routage AODV qui présente des performances insatisfaisantes à mesure que le nombre de nœuds, la mobilité des nœuds et les flux de trafic augmentent. Il est possible de tester les performances du modèle de confiance sur un protocole de routage basé sur la position tel que GyTAR, qui peut gérer des changements plus fréquents dans le réseau V-Mesh. En utilisant le routage basé sur la position, d'autres aspects de performance tels que l'évolutivité et le temps de convergence de confiance peuvent être évalués pour affiner davantage les modèles de confiance DS-Trust et MeTRO. Troisièmement, les modèles de confiance proposés ne peuvent pas se défendre contre les attaques Sybil, où les attaquants peuvent simuler plusieurs identités pour échapper à la détection ou se faire passer pour d'autres nœuds pour émettre des recommandations malhonnêtes.

Dans le cadre SA-KMP, l'unité RSU doit exister pour aider à l'établissement de clés par paires et de groupes. Cependant, cette hypothèse peut ne pas être vraie. En tant que travail futur, le cadre SA-KMP peut être renforcé pour étendre son applicabilité à un scénario de banlieue ou rural où les RSU ne sont pas omnipré-

sentes. La protection de la vie privée est une autre question qui a suscité beaucoup d'intérêt dans le milieu de la recherche mais qui n'a pas encore été abordée dans le cadre de la SA-KMP. Fondamentalement, en raison de la nature de la diffusion des communications, les véhicules à proximité du véhicule d'émission peuvent capturer et intercepter les messages pour une analyse ultérieure. En outre, dans le protocole d'accord de clé, chaque nœud est nécessaire pour échanger des informations telles que l'identifiant et la localisation GPS qui expose le protocole aux attaques de confidentialité. En tant que travail futur, le cadre SA-KMP doit tenir compte de la confidentialité dans la conception. En plus d'assurer la protection de la vie privée, les autorités doivent être en mesure de suivre les véhicules qui se comportent mal pour en déterminer la responsabilité. Dans le protocole de preuve de connaissance, la RSU peut exploiter la nature du protocole pour épuiser les ressources du véhicule en l'inondant de signatures à vérifier. En tant que travail futur, les évaluations de confiance peuvent être incluses dans le protocole de preuve de connaissance. De nombreux véhicules peuvent également lancer des attaques DDoS sur l'unité RSU, compromettant ainsi la disponibilité du service. Pour atténuer les attaques DDoS, l'unité RSU peut implémenter un compteur pour suivre le nombre de demandes similaires envoyées par chaque véhicule au cours d'une période donnée. Si la fréquence de la requête est très élevée, c'est une indication des attaques DDoS.

Enfin, il est intéressant d'explorer la possibilité de combiner des solutions non cryptographiques, c'est-à-dire des modèles MeTRO avec des solutions de cryptographie, à savoir SA-KMP. Le modèle de confiance MeTRO peut aider à sécuriser le routage contre les attaques internes égoïstes pour améliorer la capacité du réseau tandis que le SA-KMP fournit des communications efficaces et sécurisées de bout en bout entre les nœuds terminaux contre les attaques externes et internes avec des propriétés de confidentialité, authenticité et intégrité. Les clés symétriques dérivées par le composant de dérivation de clé dans l'infrastructure SA-KMP peuvent être utilisées dans le modèle d'approbation MeTRO pour réaliser le mécanisme d'authentification de l'arbre de Merkle.

Abstract

Secure communication is an integral part of message exchange in a vehicular network formed by the integration of Vehicular Ad-Hoc Network (VANET) and Wireless Mesh Network (WMN). However, this integration gives rise to node cooperation issue because of the multi-hop communications. Furthermore, traditional security solutions provided by the Public Key Infrastructure (PKI) approach may not be efficient because of the short connection times caused by the high mobility of vehicles. The goal of this thesis is to design trust models and key establishment protocols to provide a trusted and secure communication in a vehicular environment.

In trust modeling, recommendation trusts are leveraged to improve the detection time of selfish nodes in the network but, relying on recommendation trusts exposes the trust model to badmouthing and ballot-stuffing attacks. To overcome these vulnerabilities, we propose a trust model called the Dempster Shafer-Trust (DS-Trust) model, which is based on two techniques: the dissimilarity test and the Dempster Shafer Theory (DST). The dissimilarity test determines the amount of conflict between two trust records, and DST re-adjusts the weight of the recommendation trusts based on the dissimilarity results to downplay the impact of false recommendations. Numerical results show that DS-Trust model is robust against badmouthing and ballot-stuffing attacks when compared to other trust aggregation techniques such as the linear opinion pooling, subjective logic model, entropy-based probability model and regression analysis. Through NS-3 simulations, DS-trust model can mitigate selfish attacks such as blackhole and grayhole attacks.

Another problem with the trust model is that it depends on the overhearing mechanism to derive trust ratings, which is susceptible to limited transmission power and packet modification attacks that may affect the judgment of the nodes. To address these issues, we propose a novel trust model called the Merkle Tree-based with Reinforced Overhearing (MeTRO) using two techniques. First, it leverages on upstream monitoring to reinforce the overhearing observations collected from the downstream monitoring to mitigate limited transmission power attacks. Second, it incorporates an efficient Merkle-based tree authentication mechanism for detecting modified packets along a multi-hop path. Through extensive simulations, we demonstrate that MeTRO trust model can resist attacks associated with overhearing, including packet dropping attacks. Moreover, the Merkle-based tree authentication

mechanism introduced in the MeTRO trust model is scalable in terms of the authentication delay when compared to the Elliptic Curve Digital Signature algorithm (ECDSA) for verifying the authenticity of messages.

To reduce the communication costs of deploying a PKI, we propose a Secure and Authenticated Key Management Protocol (SA-KMP). The SA-KMP scheme eliminates the exchange and management of certificates and Certificate Revocation Lists (CRLs) by delegating the management of keys to each node in the network by means of distributing repositories containing the bindings of the node's identity and its corresponding public key. To reduce the high computing costs of asymmetric cryptography, the SA-KMP scheme uses symmetric keys derived based on a 3D matrix-based key agreement scheme to secure the communications. We demonstrate the efficiency of SA-KMP through performance evaluations in terms of transmission overhead, storage overhead, network latency, scalability and key generation time by comparing it to the certificate-based PKI and the Elliptic Curve Diffie-Hellman (ECDH), and Diffie-Hellman (DH) protocols. In addition, we use an automatic cryptographic protocol verifier called Proverif to prove that the key agreement protocol of the SA-KMP scheme is secure against an active attacker under the Dolev and Yao model and further show that SA-KMP scheme is secure against Denial of Service (DoS), collusion attacks and a wide range of other malicious attacks.

Keywords - *Dempster's rule of combination, information fusion, packet dropping attacks, recommendation-based trust model, reputation-based attacks, limited transmission power attacks, Merkle tree authentication mechanism, VANET, WMN, certificateless PKI, hybrid cryptosystems, Proverif, 3D grid-based key agreement*

Contents

Acknowledgements **ii**

Résumé **iii**

Abstract **xlix**

List of Figures **lvi**

List of Tables **lviii**

List of Acronyms **lix**

List of Notations **1**

1 Introduction **1**

1.1 Emergence of C-ITS 1

1.2 Background of V-Mesh Network 3

1.2.1 Network Model 3

1.2.2 Communication Model 6

1.2.3 Features 7

1.2.4 Applications 9

1.3 Motivations and Objectives 11

1.4 Contributions 12

1.5 Organization of Thesis 14

2 Related Works **17**

2.1 Attacks in V-Mesh Network 18

2.1.1 Classification of Attacks 18

2.1.2 Attacks on Security Goals 21
2.2 Challenges in V-Mesh Network 23
2.2.1 Routing Issues 24
2.2.1.1 Blackhole Attack 24
2.2.1.2 Grayhole Attack 25
2.2.1.3 Packet Modification Attack 25
2.2.2 Key Management Issues 26
2.2.2.1 Traditional PKI 26
2.2.2.2 Issues of PKI 27
2.3 Review of Trust-based Routing 28
2.3.1 Components of Trust Model 28
2.3.1.1 Trust Monitoring 29
2.3.1.2 Trust Aggregation 30
2.3.1.3 Decision Making 31
2.3.2 Survey of Trust-based Schemes 32
2.3.2.1 Reputation-based Approach 33
2.3.2.2 Credit-based Approach 35
2.3.2.3 Discussions 36
2.3.3 Survey of Trust Aggregation Techniques 37
2.3.3.1 Linear Opinion Pooling 37
2.3.3.2 Entropy-based Probability Model 38
2.3.3.3 Subjective Logic Operators 39
2.3.3.4 Regression Analysis 40
2.3.3.5 Discussions 41
2.4 Review of Key Management 43
2.4.1 Symmetric Cryptography 43
2.4.1.1 Key Distribution Schemes 44
2.4.2 Asymmetric Cryptography 47
2.4.2.1 Certificate-less PKI 47
2.4.2.2 Reduced CRL Checking 49
2.5 Conclusion 51

3 An Unbiased Trust Model 53
3.1 Preliminaries 53

3.1.1 Design Goals 54
3.1.2 DST Background 55
3.1.2.1 Basic Probability Assignment 55
3.1.2.2 Belief Function and Plausiblity Function 56
3.1.2.3 Dempster's Rule of Combination 57
3.2 DS-Trust Model 57
3.2.1 Monitoring Module 58
3.2.2 Feedback Module 59
3.2.3 Correlation Module 60
3.2.4 Fusion Module 60
3.2.5 Decision Module 63
3.3 Security Analysis 64
3.3.1 Experimental Setup 64
3.3.2 Resiliency to Badmouthing Attacks 65
3.3.3 Resiliency to Ballot-stuffing Attacks 67
3.4 Performance Analysis 68
3.4.1 Simulation Environment 68
3.4.2 PDR Performance 71
3.4.3 NRO Performance 74
3.4.4 Effects of Mobility 77
3.4.5 Throughput Performance 78
3.4.6 Computational Complexity 80
3.5 Conclusion 81

4 Authenticated Trust Model with Reinforced Overhearing 83
4.1 Preliminaries 84
4.1.1 Message Commitment Value 84
4.1.2 Mobility Profiling 86
4.2 MeTRO Trust Model 88
4.2.1 Downstream Monitoring Module 89
4.2.2 Upstream Monitoring Module 90
4.2.3 Direct Trust Evaluation Module 91
4.2.4 Recommendation Module 92
4.2.5 Fusion Module 93

4.2.6 Decision Module 95
4.3 Security Discussion 96
4.3.1 Eavesdropping Attacks 96
4.3.2 Message Replay Attacks 96
4.3.3 Packet Modification Attacks 97
4.3.4 Repudiation Attacks 98
4.4 Efficiency Analysis 98
4.4.1 Generation Time of Commitment Value 98
4.4.2 Authentication Delay 100
4.5 Performance Evaluation 101
4.5.1 Simulation Setup 101
4.5.2 Performance under Limited Transmission Power Attacks . . . 103
4.5.3 Performance under Packet Modification Attacks 104
4.5.4 Performance under Packet Dropping Attacks 105
4.5.4.1 Resiliency to Blackhole Attacks 105
4.5.4.2 Resiliency to Grayhole Attacks 107
4.6 Conclusion 109

5 Secure and Authenticated Key Management Protocol 111
5.1 Introduction 111
5.1.1 Enhancements to PKR 113
5.1.1.1 DoS Mitigation 113
5.1.1.2 Collusion Mitigation 115
5.1.1.3 Distribution of Public Keys 115
5.1.2 Enhancements to 3D Grid-based Key Distribution 116
5.1.2.1 Node Capture Attacks Mitigation 116
5.1.2.2 DoS Mitigation 117
5.2 SA-KMP Scheme 117
5.2.1 System Setup Module 117
5.2.2 Public Key Request Module 120
5.2.3 Key Agreement Module 123
5.2.3.1 Pairwise Keys Agreement 123
5.2.3.2 Group Keys Agreement 127
5.2.4 Key Derivation Module 130

5.2.5 Revocation Module 131
5.3 Security Analysis 133
5.3.1 Classical Attacks 133
5.3.2 Formal Verification by ProVerif 138
5.3.2.1 Modeling of Key Agreement Module 139
5.3.2.2 Verification Results 146
5.4 Performance Analysis 146
5.4.1 Transmission Overhead 147
5.4.2 Storage Overhead 148
5.4.3 Latency Analysis 149
5.4.4 Scalability 153
5.4.5 Key Generation Time 157
5.4.6 Computational Complexity 160
5.5 Conclusion 161

6 Conclusions **162**
6.1 Summary 162
6.2 Recommendation for Future Work 165
6.2.1 Trust Models Extensions 165
6.2.2 SA-KMP Extensions 166

List of Publications **168**

Bibliography **169**

List of Figures

1.1 C-ITS communication technologies. 2
1.2 Components of a V-Mesh network. 4
1.3 Overview of our work. 14

2.1 Classification of attacks. 19
2.2 Classification of challenges 23
2.3 Traditional PKI architecture. 26
2.4 Components of trust model. 29
2.5 Classification of trust-based schemes 32
2.6 Traffic monitoring procedure 34
2.7 Classification of key management solutions 43

3.1 DS-Trust model. 58
3.2 Trust aggregation. 65
3.3 Trust value as a function of badmouthing recommenders. 66
3.4 Trust value as a function of ballot-stuffing recommenders. 67
3.5 Simulation topology for infrastructure-based V-Mesh. 69
3.6 Simulation topology for hybrid-based V-Mesh. 69
3.7 PDR in an infrastructure-based V-Mesh under blackhole attacks. . . . 71
3.8 PDR in a hybrid-based V-Mesh under blackhole attacks. 72
3.9 PDR in an infrastructure-based V-Mesh under grayhole attacks. . . . 72
3.10 PDR in a hybrid-based V-Mesh under grayhole attacks. 73
3.11 NRO in an infrastructure-based V-Mesh under blackhole attacks. . . 74
3.12 NRO in a hybrid-based V-Mesh under blackhole attacks. 75
3.13 NRO in an infrastructure-based V-Mesh under grayhole attacks. . . . 76
3.14 NRO in a hybrid-based V-Mesh under grayhole attacks. 76
3.15 PDR in a hybrid-based V-Mesh under varying speed. 77

3.16 NRO in a hybrid-based V-Mesh under varying speed. 78
3.17 Throughput in an infrastructure-based V-Mesh. 79
3.18 Throughput in a hybrid-based V-Mesh. 79

4.1 Merkle tree construction with 8 data blocks. 85
4.2 Mobility pattern of vehicle A. 87
4.3 Proposed MeTRO trust model. 89
4.4 Delay for generating the Merkle root. 99
4.5 Delay for authenticating a data block. 99
4.6 Authentication delay of different schemes as a function of message load. 101
4.7 PDR of different schemes under limited transmission power attacks. . 103
4.8 PDR of different schemes under packet modification attacks. 104
4.9 PDR of different schemes as a function of blackhole vehicles. 106
4.10 E2E delay of different schemes as a function of blackhole vehicle. . . . 106
4.11 NRO of different schemes as a function of blackhole vehicles. 107
4.12 PDR of different schemes under different dropping rates. 108
4.13 E2E delay of different schemes under different dropping rates. 108
4.14 NRO of different schemes under different dropping rates. 109

5.1 Difference between Schnorr and DSA signing process 114
5.2 SA-KMP framework. 118
5.3 Authentication process in the public key request module. 121
5.4 Exchange of keying materials for establishing pairwise keys in V2I. . . 124
5.5 Exchange of keying materials for establishing group keys in V2V. . . 128
5.6 Authentication delay under different scenarios. 134
5.7 Proverif verification results. 146
5.8 Transmission overhead as a function of messages. 148
5.9 Latency of SA-KMP, certificate-based PKI and PKR schemes. 150
5.10 Scalability in terms of transmission overhead and storage overhead. . 155
5.11 Scalability in terms of latency experienced by a user. 157
5.12 Key generation time as a function of network size. 158
5.13 Key generation time as a function of plane equations. 159

List of Tables

2.1 Description of attacks 19
2.2 Attacks on security goals 22
2.3 Summary of trust-based schemes and aggregation techniques 41
2.4 Summary of key management solutions 50

3.1 Aggregation of evidence source E_1 and evidence source E_2 62
3.2 Simulation parameters 70

4.1 Mobility profile of vehicle A in each RSU 88
4.2 Aggregated profile of vehicle A in each RSU 88
4.3 Simulation parameters 102

5.1 Contents of RSU-PFD 119
5.2 Contents of VPFD 119
5.3 Authentication delays of the different components for different schemes 134
5.4 Simulation settings 134
5.5 Comparison of transmission overhead for various schemes 147
5.6 Comparison of storage overhead for various schemes 148
5.7 Average timings for calculating processing delay 151
5.8 Average timings for calculating propagation delay 152
5.9 Network latency of different schemes 153
5.10 Transmission overhead for different schemes for a single user 154
5.11 Storage overheard for different schemes in bytes (**B**) 154
5.12 Various parameters for calculating the latency 156
5.13 Simulation settings for the first experiment 157
5.14 Simulation settings for second experiment 159
5.15 Number of cycles for C implementation of SA-KMP 160

List of Acronyms

ANN	Artificial Neural Network
AODV	Ad hoc On-Demand Distance Vector
bpa	basic probability assignment
CA	Certificate Authority
CAD	Channel Aware Detection
CBR	Constant Bit Rate
C-ITS	Cooperative-Intelligent Transportation Systems
CONFIDANT	Cooperation of Nodes: Fairness in Dynamic Ad-hoc NeTworks
CORE	COllaborative REputation
CRL	Certificate Revocation List
C-VeT	Campus-Vehicular Testbed
DDoS	Distributed Denial of Service
DH	Diffie-Hellman
DOME	Diverse Outdoor Mobile Environment
DoS	Denial of Service
DSA	Digital Signature Algorithm
DSRC	Dedicated Short Range Communications
DST	Dempster Shafer Theory

DS-Trust	Dempster Shafer-Trust
E2E	End-to-End
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature algorithm
ECN	Electronic Chassis Number
ELP	Electronic License Plate
EM	Expectation-Maximization
EMAP	Expedite Message Authentication Protocol
GPS	Global Positioning System
GyTAR	Greedy Traffic Aware Routing
HMAC	Hashed Message Authentication Code
HSM	Hardware Security Module
HWMP	Hybrid Wireless Mesh Protocol
I2I	Infrastructure-to-Infrastructure
IBC	ID-Based Cryptography
IoTs	Internet of Things
IR	Infrared
JSD	Jensen Shannon Divergence
LTE	Long Term Evolution
MANET	Mobile Ad-Hoc Network
MeTRO	Merkle Tree-based with Reinforced Overhearing

MPR	Multipoint Relay
NRO	Normalized Routing Overhead
OBU	On Board Unit
OEM	Operator Equipment Manufacturer
P2I	Pedestrians-to-Infrastructure
PDR	Packet Delivery Ratio
PKC	Public Key Cryptography
PKI	Public Key Infrastructure
PKR	Public Key Regime
PPS	Payment Punishment Scheme
QoS	Quality of Service
RERR	Route Error
RREP	Route Reply
RREQ	Route Request
RSS	Received Signal Strength
RSU	Roadside Unit
RSU-PFD	RSU Public File Directory
SA-KMP	Secure and Authenticated Key Management Protocol
SORI	Secure and Objective Reputation-based Incentive
SP	Service Provider
TA	Trusted Authority
TMS	Traffic Management System
UMTS	Universal Mobile Telecommunications System

V2I	Vehicle-to-Infrastructure
V2P	Vehicle-to-Pedestrians
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
VANET	Vehicular Ad-Hoc Network
VCG	Vickrey-Clarke-Groves
V-Mesh	Vehicular-Mesh
VPFD	Vehicle Public File Directory
WiFi	Wireless Fidelity
WiMax	Worldwide Interoperability for Microwave Access
WMN	Wireless Mesh Network

List of Notations

(e_i, S_i)	Signature pair issued by entity i
(i_i, j_i, k_i)	3D location of entity i
$Dist_{LB}, Dist_{UB}$	Lower bound (LB) and upper bound (UB) of the distance that derived keys stay valid.
E	Elliptic Curve
exp_{LB}, exp_{UB}	Lower bound (LB)and upper bound (UB) of the expiry time of derived keys
F_p	Finite field of p elements where p is a prime number
G	Generator of a group or subgroup of $E(F_p)$
h	Cofactor where $h \leq 4$
$h : \{0, 1\}^*$	Secure hash algorithm
H_1, H_2, H_3	Secure hash functions along each axis (x, y, z) in 3D space
k_i^- / k_i^+	Private key/Public key of entity i
$K_{GRP,i}$	Group key initiated by entity i
$K_{i,j}$	Pairwise key between entity i and entity j
$LOCM_i$	GPS location of entity i
$m \times m \times m$	Dimensions of 3D space
n	Order of the generator G

N_i	Nonce value of entity i used in key agreement module
R_i, C_i	Cryptographic nonce and commitment value of entity i
$Sig_{k_i^-(M)}$	Signature of message M signed by private key of entity i
T_i	Timestamp of entity i used in key agreement module
v_i	Hash value issued by entity i as proof
V_{avg}	Average speed of vehicle in m/s.
x_p, y_p	The x and y coordinates of point P in affine representation
$M_{k_i^+}$	Message M encrypted by public key of entity i

Chapter 1

Introduction

1.1 Emergence of C-ITS

Vehicular Ad-Hoc Network (VANET) is a self-organizing network formed by a collection of moving vehicles [1, 2]. Each vehicle communicates with one another using Dedicated Short Range Communications (DSRC) technology that operates in the 5.9 GHz frequency band [3, 4]. The goal of VANETs is to improve the public road safety and the efficiency of the transportation systems [2, 5, 6]. But recently, due to advances in communication technologies and the proliferation of wireless devices, many car manufacturers have begun to install sensors, Global Positioning System (GPS), cameras and other communication modules on their vehicles to make them “smarter". This means that vehicles are not only able to communicate among themselves, they are also able to interact intelligently with the surrounding environment. These activities eventually led to the development of Cooperative-Intelligent Transportation Systems (C-ITS) [7, 8, 9] which aims to harmonize communication technologies to enable various communication nodes to interact seamlessly.

In C-ITS, all the elements of the transport chain ranging from the public transport down to the individual road users such as the pedestrians, cyclists, and motorcyclists are connected together. In addition, various communication technologies can be supported to provide continuous and seamless connectivity. These technologies such as cellular, Worldwide Interoperability for Microwave Access (WiMax), Wireless Fidelity (WiFi), Infrared (IR) etc. can be classified into long range, medium range and short range coverage as shown in Figure 1.1. With the support of these

access technologies, C-ITS aims to extend the reach of classical VANET applications beyond safety and traffic management applications. In this regard, it is envisaged that value-added services such as point of interest notification, electronic commerce, media downloading and Internet provisioning services such as in-car entertainment applications can be provided to further enhance the comfort level of a road user [8].

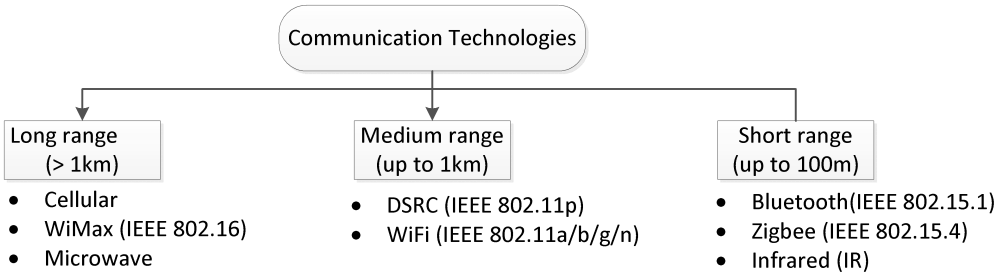

Figure 1.1: C-ITS communication technologies.

In fact, several initiatives [10, 11, 12, 13, 14] have been developed to integrate VANET with cellular networks such as the Universal Mobile Telecommunications System (UMTS) and the Long Term Evolution (LTE). WiMax networks have also been proposed for vehicular communications to tackle the coverage problem in C-ITS [15]. There have also been several testbeds deployed by research institutions to study the performance of integrating VANET with WiFi based technology. Particularly, UCLA has deployed a Campus-Vehicular Testbed (C-VeT) [16] where the interface between the vehicles and the Internet is provided by a mesh technology [17, 18]. Another testbed called the Diverse Outdoor Mobile Environment (DOME) [19] has been developed using 40 buses with a mesh network built over the city of Amherst to provide research and development work on routing, power management, and system design. More recent testbed include the HarborNet [20] which is deployed in the sea port of Leixoes in Portugal to allow for cloud-based code deployment, remote network control, and distributed data collection from the vehicles and Roadside Units (RSUs).

Among the various communication technologies, mesh networking has emerged as a promising wireless technology from which to extend the data access services of a pure VANET. In this regard, the use of Wireless Mesh Networks (WMNs) has the

added advantage of being highly interoperable with VANETs to support single-hop and multi-hop communications. This is in contrast to the cellular-based technology currently used to support only one-hop communication. Moreover, cellular networks are heavily dependent on the deployment and maintenance of expensive fixed infrastructures, which hinders scalability. On the other hand, the wireless mesh device operates in an self-organizing mode, which is less expensive to deploy and can be used to extend the network coverage easily. Other benefits of WMN include (1) Rapid deployment with lower-cost backhaul, (2) Easy to provide coverage in hard-to-wire areas, (3) Self-healing, resilient, extensible, (4) Greater range due to multi-hop forwarding, (5) Higher bandwidth due to shorter hops, (6) Better battery life due to lower power transmission.

For the above reasons, we consider in our work a C-ITS network that is formed by integrating a VANET and a WMN together. In the remainder of this thesis, we call this integration a Vehicular-Mesh (V-Mesh) network for ease of exposition.

1.2 Background of V-Mesh Network

In the following, we present the architectural view of the V-Mesh network. We discuss the communication model and analyze several key features of the network. Finally, we describe some applications that can be supported by V-Mesh network to highlight its high versatility.

1.2.1 Network Model

Figure 1.2 shows the architecture of a V-Mesh network where the WMN is deployed as an overlay to extend the data access services of a VANET. As seen in the figure, the architecture comprises of four ITS subsystems [21, 22] defined as follows:

- **Central subsystem** refers to the public authorities such as the Traffic Management System (TMS), the Certificate Authority (CA) and the Service Providers (SPs) etc. whose role is to provide management, administrative, and support functions for the transportation system throughout the region. For example, the TMS collects real-time traffic updates, analyzes them and implements response strategies to ensure safe and efficient use of the road network. The

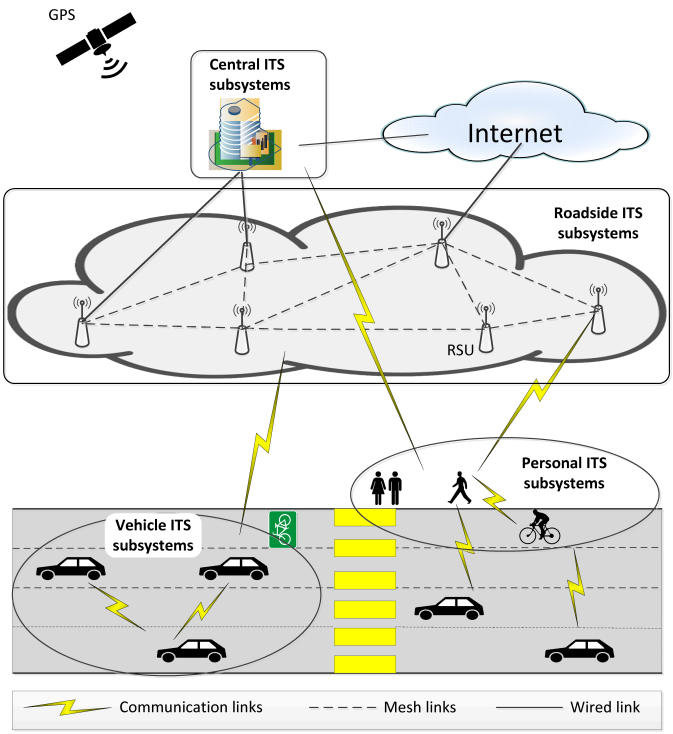

Figure 1.2: Components of a V-Mesh network.

CA authenticates the identity of an individual and creates a digital certificate that binds the user's identity and its public key together. It also maintains a Certificate Revocation List (CRL) for tracking invalid digital certificates and periodically disseminates the CRL to inform other correspondent nodes of the revocation. On the other hand, the SP collects traffic information from different sources and redistributes it to other SPs for further data processing. Other tasks undertaken by the SP include delivering traveler information to subscribers and provide basic advisories, real time traffic condition, and video-on-demand etc. to improve the road user experience. In general, the central subsystem has a high computing power to implement various security mechanisms. Therefore, the TMS, CA and SPs are fully trusted and tamper resistant.

- **Roadside subsystem** comprises of RSUs which are stationary devices installed at the traffic lights, gantries, or at critical junctions. They are equipped with two radios [20] to support the IEEE 802.11s [23] and the IEEE 802.11p [3, 24] wireless technologies. The IEEE 802.11s allows the RSUs to establish

wireless mesh links among themselves to support mesh networking while the IEEE 802.11p radio helps to bridge communication with other vehicles and pedestrians in the network. The RSUs may aggregate inbound messages, accept outbound messages and help to relay messages to and from the central subsystem and the vehicle subsystem. Some of the RSUs are connected to the central subsystem via a secure wired network while some of them are configured as gateways to connect to the Internet. Additionally, each RSU is equipped with a Hardware Security Module (HSM) [25, 26] to store the cryptographic keys used for communications. The HSM is assumed to be tamper resistant, such that any tampering will cause it to self-destruct. Since the RSUs are not resource-constrained and are managed by the central subsystem, they are generally trusted.

- **Vehicle subsystem** is made up of vehicles which can be classified into two types: privately owned or publicly owned. The public vehicles refer to public buses, trams, subways etc. under the jurisdiction of the national transport authority. Vehicles in this domain travel at different speeds in the network depending on the road conditions and the vehicle type. Each vehicle is pre-installed with an On Board Unit (OBU) having similar hardware as an RSU except that the OBU is only equipped with an IEEE 802.11p radio for communicating with the RSUs i.e. the roadside subsystem. Besides that, the OBU in each vehicle collects information about the vehicle's dynamics such as the direction of heading and speed and share this information with the other vehicles and the RSUs. Each vehicle is also equipped with a GPS receiver with accuracy up to 5 meters [27] to facilitate location-based services. In addition, each vehicle is equipped with a HSM to manage the cryptographic keys. Since the vehicles are privately-owned and constitute the primary population of the V-Mesh network, the vehicle can misbehave and carry out a series of attacks.

- **Personal subsystem** refers to portable devices that are not part of the vehicle communication system. Rather, they are hand-held devices such as smart phones, tablets, personal navigation devices that are carried by pedestrians and cyclists in the network. Users of these devices typically obtain ITS related services through the various “apps" installed on them. In terms of communication

behaviors, these devices use cellular or WiFi technology to communicate with the central subsystem and the roadside subsystem respectively. In our work, the communication between the personal devices and the central subsystem is beyond the scope of this thesis.

1.2.2 Communication Model

V-Mesh communications can be generalized into Vehicle-to-Everything (V2X) which denotes that a vehicle is able to interact with a diverse range of wireless devices [9]. These devices, as depicted in Figure 1.2 comprise of the RSUs, pedestrians, cyclists and other vehicles etc. Depending on who is the receiving end of a communication channel, V2X can be categorized into three main types i.e. Vehicle-to-Infrastructure (V2I), Vehicle-to-Vehicle (V2V) and Vehicle-to-Pedestrians (V2P) including other specific communication types such as the Infrastructure-to-Infrastructure (I2I) and the Pedestrians-to-Infrastructure (P2I) communication which we describe below:

- **V2I** defines the communication between a vehicle and an RSU or vice-versa. The aim is to provide environment sensing so as to improve road safety and minimize environmental impacts. This type of communication is also used to support in-car entertainment system such as video streaming to enhance the driving experience.

- **V2V** defines the communication between two vehicles on the road where vehicles exchange information such as warning messages to alert one another of accidents or collisions ahead. V2V communication is also used for specialized groups such as police department or emergency vehicles to facilitate group communications. Another use of V2V communication is platooning applications [28] where the vehicles exchange coordination messages to facilitate cooperative driving.

- **V2P** defines the communication between a vehicle and a broad set of pedestrians such as cyclists, wheelchair users including but not limited to other mobility device users, and passengers embarking and disembarking buses etc. This type of communication is useful in that vehicles attempting to make a turn at an intersection are alerted to the presence of a pedestrian crossing the

road. Similarly, pedestrians can receive warning messages via V2P communication to avoid potential dangers during crossing.

- **P2I** defines the communication between a pedestrian and an RSU or vice-versa. By interacting with the infrastructure, the pedestrian can interpret and predict vehicular movement with higher accuracy than using V2P communication where the pedestrian device can only communicate with the vehicle within a close range. Pedestrians can also communicate with the RSUs to obtain real-time traffic updates to avoid commuting delays.

- **I2I** defines the communication between two RSUs within the roadside subsystem. It is used whenever a message needs to be sent to the TMS that can only be reached via another RSU several hops away. Another use case of I2I includes the controlling of traffic signals along a path to give way to emergency vehicles. In the event of a disaster, I2I can also help to maintain communication and provide redundancy when communication with the TMS fails.

We note that, regardless of the type of communications, information is usually delivered to the destination in a multi-hop manner. In addition, these communications are protected using asymmetric cryptography and digital certificates as specified in [29]. To generate the signatures, the Elliptic Curve Digital Signature algorithm (ECDSA) based on the P-224 or P-256 curves is assumed.

1.2.3 Features

Even though V-Mesh network is a heterogeneous network, it inherits many features from the classical VANET, which we describe below:

- **Varied mobility:** V-Mesh network is characterized by a varying mobility pattern, mainly due to the different types of communication nodes in the network. For example, the walking speed of a pedestrian is usually between 4 km/h and 6 km/h. For cyclists or pedestrians taking other forms of motorized transport, the average speed can go up to 30 km/h. Similarly, the speed of a moving vehicle can vary widely depending on the driving environment and the traffic conditions. As an example, the maximum speed of a vehicle is capped

at 60 km/h in an urban environment whereas on the highway, a vehicle speed can travel at a speed of 100 km/h or more. For these reasons, the mobility is always changing and is undergoing constant changes.

- **Highly dynamic topology:** Due to the constant changes in mobility, vehicles including pedestrian devices can leave and join the network very rapidly. Therefore, V-Mesh network is highly dynamic and subject to frequent topology changes. The topology is further affected by the driving environment. For instance, in the urban scenario where the road network is much denser than the highway model and is characterized by many intersections, the communication nodes are more likely to travel in different directions, resulting in unpredictable topology changes.

- **Short connection times:** As a result of the dynamic topology and the high mobility of the vehicles, the connection time between any two vehicles is very short which makes it very challenging to establish a stable communication. Even if a communication link has been established, there are frequent disconnections when the vehicles move away from each other. Consequently, it is difficult to maintain reliable communication and membership in a group.

- **Multi-hop communications:** Although DSRC technology can support a transmission range up to 1000m [3], shorter transmission ranges are generally preferred for better reliability. As a result of the shorter communication range, the V-Mesh network has to rely on the cooperation of others to send the data packets to the intended destination. This feature has the benefit of extending the network coverage and throughput including the battery life of hand-held devices. Unfortunately, it also exposes the network to a wide range of attacks.

- **Hard delay constraints:** Most C-ITS applications have strict delay constraints to meet application requirements. For example, in a cooperative forward collision warning application, messages must be communicated to other vehicles within a time period of 100ms to warn the drivers of an accident [9]. As a result, latency is an important criteria when managing safety-related applications. For a more comprehensive list of applications and their latency requirements, we refer the readers to [30].

- **Unbounded network size:** Since V-Mesh networks must support a wide range of communication technologies involving multiple stakeholders, this means that V-Mesh deployments are typically larger than classic VANETs and are not limited to a particular geographical area [31]. Furthermore, due to the higher application support, the network and traffic density are higher compared to pure VANETs.

1.2.4 Applications

In terms of application scenarios, C-ITS applications can be categorized into three types namely, road safety, traffic management, and value-added applications [2, 9, 32, 33, 34]. In the first type, the aim of road safety applications is to broaden the perception of road users so that they can make better-informed judgments on the road to improve road safety. For the second type of application, the main focus is to reduce the environmental impact of transportations. This can be achieved by implementing services such as lane merging assistance to minimize the possibility of congestions. Finally, value-added applications aim to provide comfort and convenience services for the road users to make their journey more pleasant. In what follows, we list several applications under each category.

- **Road safety applications:** Some examples of road safety applications are pedestrian crossing warning, collision avoidance, and emergency vehicle warning systems. To realize the pedestrian crossing warning system, sensors are installed onto the RSUs to detect the presence of pedestrians. If a pedestrian is detected, the RSU will broadcast a warning message to the vehicles in that particular region to inform them to slow down. A similar mechanism is used in the collision avoidance systems wherein the surrounding RSUs collect information to assess the possibility of potential hazards and then a message is sent to notify the neighboring vehicles. The difference between the pedestrian crossing warning and the collision avoidance is that in the latter, the warning message is relayed to a larger group of vehicles so that en route vehicles a few kilometers away are also aware of the hazard. In the emergency vehicle warning system, the main idea is for the emergency vehicles such as the ambulance to trigger the nearby traffic lights to green so as to reduce the emergency

response times. For a list of other road safety applications, we refer readers to [33] for more details.

- **Traffic management applications:** There is another class of vehicular applications that is closely related to road safety applications. However, the focus is not on safety but on the overall efficiency of the transport system. An example is the electronic toll collection where vehicles are charged whenever they pass by certain road segments. The idea is to encourage the vehicles to change their travel pattern or time of travel so as to avoid traffic congestion on heavily used roads. Since it is an automated computer system rather than a manual billing system, any human errors that arise from the manual billing are eliminated, thereby improving accuracy and accountability. Another example of a traffic management applications is the speed limit notification system whereby some RSUs at strategic locations are retrofitted with a tracking device to monitor the speed of the vehicles. If over speed occurs, the respective RSU will unicast a message to raise awareness to the drivers. In some cases, the system can also work against the drivers to catch traffic offenders.

- **Value-added applications:** In addition to the above applications, another emerging vehicle application is the value-added applications, which is the result of the C-ITS initiative implemented with the support of various communication technologies. These applications include remote vehicle diagnostic and maintenance services as well as Internet access such as content downloading, weather information and notifications of the nearest point of interests. In general, any application designed to improve the driving experience falls into this category. In order to access these value-added services, the vehicle usually contacts a nearby RSU, which will then help relay the request to the remote SP for action. The SPs can also take a more proactive approach, such as in vehicle diagnostic and maintenance application, by collecting in-car sensors information, collating them, and then automatically pushing the software updates to the drivers'vehicle. [9].

1.3 Motivations and Objectives

Due to the interesting features of the V-Mesh network such as the dynamic topology and the multi-hop communications, cooperation among nodes is more difficult to establish than in traditional networks. Some nodes can suppress forwarding in order to save their resources. As a result of these misbehaviors, the network suffers from significant throughput degradation. In the worst case, the discarded messages may cause the TMS to make wrong decisions and lead to traffic chaos or even lead to traffic accidents.

In order to address the node selfishness problem, various trust models [35] have been proposed to distinguish well-behaved node and selfish ones. These models calculate the trust ratings based on direct interactions including indirect observations from others in the network. The problem with collecting indirect observations is that the node as a recommender may be dishonest and lie about the trust ratings to influence the trust value of others. To this end, our first objective in this thesis is to

Objective 1:

"Design a trust model that can mitigate the effects of false recommendations".

In addition to the problem of false recommendations, using overhearing to monitor the behaviors of nodes is unreliable and inaccurate according to [36]. Specifically, an attacker can control the transmission power so that the packet is overheard but not passed to the next hop. Moreover, in a network where thousands of mobile vehicles are roaming at different speeds, the overhearing may not be effective due to the inevitable collisions and interferences. An attacker can also modify the contents of a packet before forwarding, which cannot be detected if a node fails to overhear. Consequently, the trust rating based on overhearing is highly ambiguous. Thus, the second objective of this thesis is to

Objective 2:

"Develop a trust model to improve the overhearing process and detect message alterations during packet forwarding.

As with most wireless networks, V-Mesh network relies heavily on the wireless exchange of information which are susceptible to various security attacks. In order to provide communication security, the IEEE 1609.2 [29] and the ETSI TS 102 940 standards [33] have proposed using a Public Key Infrastructure (PKI) to manage the private/public keys of all users. In this setup, digital signatures with digital certificates are used to protect the communication. The problem with PKI is the high latency involved in validating the node's digital certificates including the verification of digital signatures. Due to the frequent disconnection and short duration of communications in the V-Mesh network, the high latency of the PKI operation may not be able to support time-critical C-ITS applications, especially in safety-related applications. Thus, our last objective is to

Objective 3:

"Develop an alternative approach to PKI that can meet the latency requirements without compromising security.

In summary, the main purpose of this thesis is to design security protocols in a V-Mesh network to achieve trusted and secure communications in the most efficient way, without selfishness and malicious attacks at the highest possible security level.

1.4 Contributions

Our contributions can be divided into three parts. First, we propose a Dempster Shafer-Trust (DS-Trust) model to address the issues of receiving false recommendations. The DS-Trust model is based on the Dempster Shafer Theory (DST) whereby we exploit its ability to quantify uncertainty as a means of aggregating direct and indirect trusts. In our model, every incoming recommendation trusts are compared with the direct trust held by a node. The difference between the two trust records indicates the amount of conflict which DST classifies as uncertainty. Based on this value, the weight of the indirect trust is re-adjusted before it is combined with the direct trust to form the aggregated trust value. Through modeling uncertainty with DST, our trust model is able to alleviate the effects of bad recommendations and can withstand a greater number of malicious recommenders. Our trust model is also able to overcome the packet dropping attacks launched by the selfish nodes.

Second, we propose a Merkle Tree-based with Reinforced Overhearing (MeTRO) trust model which we develop using the DS-Trust model as a base. We introduce two additional techniques to overcome the limitations of overhearing and to improve the resiliency against packet modification attacks. In the first technique, we exploit the upstream monitoring to reinforce the overhearing results gathered from the downstream monitoring. Through the use of upstream monitoring, we demonstrate that limited transmission power and partial dropping attacks are mitigated. Furthermore, we propose to weigh the recommendation trusts based on the driving pattern of the vehicles to improve the quality of the recommendations. In the second technique, we incorporate a Merkle-based tree authentication mechanism into MeTRO trust model to enable each recipient of a packet to verify the authenticity and integrity of the message efficiently. Thus, in this model, the trust rating of a node is not only determined by the node's forwarding behavior but it is also a function of the node's integrity during packet forwarding thereby improving the trust representation and quantification.

Third, we design a Secure and Authenticated Key Management Protocol (SA-KMP) to overcome the complexities of deploying a PKI. To eliminate the complex and the high costs of certificate management, we decentralize the certificate management of a PKI by distributing public directories containing the certified public keys to each vehicle and RSU. The directories that are issued to the vehicles and the RSUs are different in that, the vehicles store the RSUs' public keys while the directory held by the vehicles contains only the certified public keys of the RSUs. In this way, communications involving different communication nodes i.e. V2I can commence directly without exchanging certificates. However, in a V2V communication, a vehicle would still need to contact an RSU to request for the public key of the target vehicle which exposes this method to Denial of Service (DoS) attacks. To mitigate the DoS attacks, we propose an authentication scheme whereby the vehicles and the RSU take turns exchanging signatures for verification before the requested public key is issued. Through mutual authentication, requesting vehicles have no incentive to misbehave as it is also going to cost them huge computational costs.

To further reduce the dependence on asymmetric cryptography, we develop a key agreement protocol to negotiate common keys for use by symmetric cryptography to secure the messages during communications. The method we adopt for establishing

the symmetric keys is based on [37, 38] where each node solves a system of linear plane equations to determine the location of the shared key in 3-D space. However, instead of preloading the shared keys onto the vehicles and RSUs prior to deployment as per the original proposal, we introduce a series of hashing functions using a nonce value as an input to derive the shared keys dynamically. The advantage of using dynamic keys instead of static preload keys is resiliency against node capture attack. If a set of keys held by a node is compromised, none of the other nodes would be at risk since every node has different and unique keys. The second advantage of dynamic keys is that there is no need to update or refresh the key space periodically which saves a lot of communication overhead.

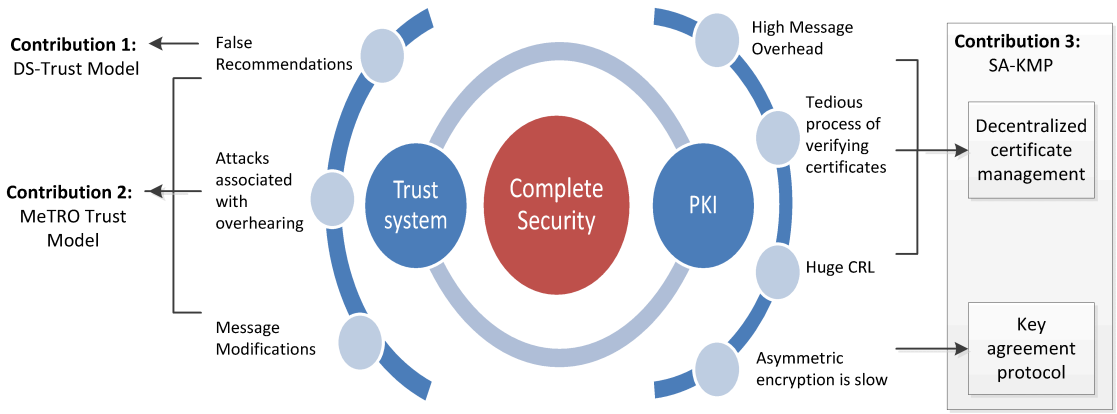

Figure 1.3: Overview of our work.

To summarize, Figure 1.3 illustrates a list of our contributions where we address the security of V-Mesh networks from two perspectives that are, trust model and key management system. According to Figure 1.3, we identify the various drawbacks concerning these two popular approaches and emphasize which of these drawbacks are being addressed by DS-Trust model, MeTRO trust model, and SA-KMP respectively.

1.5 Organization of Thesis

The rest of the thesis is organized as follows:

Chapter 2 provides a review of both trust model and key management system.

In the former, we give a brief introduction about trust models and discuss how the existing trust models handle the indirect trusts and aggregate them in the process. We also present several overhearing-based models and discuss how attacks related to overhearing can subvert the trust evaluation process. In the latter, we review the related works that were proposed in the literature to overcome the complexities of deploying PKI and discuss their limitations.

Chapter 3 introduces the DS-Trust trust model. We describe the concept and properties of DST and demonstrate how DST is incorporated into the trust model to mitigate the effects of false recommendations. We illustrate how trust rating is collected, computed and aggregated using the overhearing technique and the DST in detail. Extensive simulations and numerical analysis are conducted to demonstrate the effectiveness of the DS-Trust trust model under the influence of false recommendations and selfish attacks.

Chapter 4 presents the MeTRO trust model which is an extension of the DS-Trust trust model to overcome the limitations of overhearing and the lack of packet integrity during forwarding. First, we introduce upstream monitoring to improve the trust evaluation. We illustrate how a node can rely on observations gathered from nodes that are two hops away to reinforce its own overhearing results. Second, we describe the operation of a Merkle tree authentication mechanism and discuss how it can be employed in the trust model to achieve packet correctness and freshness during forwarding. We show by simulations that MeTRO trust model can mitigate attacks related to issues of overhearing and detect packet modification attacks efficiently to achieve network performance gain.

Chapter 5 proposes the SA-KMP framework that makes use of two concepts to reduce the latency and complexity of deploying a PKI. More specifically, we discuss how complex certificate management can be eliminated by issuing public directory with up-to-date public keys to each vehicle and RSU in the network. We also introduce a novel method of establishing symmetric keys as an alternative to asymmetric cryptography to secure communication messages. We further propose countermeasures to prevent DoS and provide strong resilience against node capture attacks in SA-KMP. Extensive simulations are conducted to illustrate the security properties, scalability and efficiency of the SA-KMP framework when compared to the certificate-based PKI scheme.

Finally, Chapter 6 concludes this thesis and points out some future work. Particularly, we discuss several extensions to our proposed trust models to improve the detection capability in a lossy environment and including defense against Sybil attacks. In SA-KMP, we highlight the need for privacy protection including the establishment of symmetric keys without the RSU.

Chapter 2

Related Works

A V-Mesh network is a wireless network formed by two types of networks namely, WMN and VANET. It not only inherited the good characteristics of the two networks, but also inherited from these networks a number of inherent security vulnerabilities. Understanding these vulnerabilities makes it critical to design robust and efficient security solutions for V-Mesh networks. To begin with, we start with an overview of security attacks in a V-Mesh network in section 2.1 and then discuss the challenges of a V-Mesh network in section 2.2. In particular, we detail the routing and key management challenges in V-Mesh networks. We then list possible attacks in the routing process and the consequences of these attacks. In terms of key management issues, we point out several performance issues in deploying PKIs to motivate our study.

Cryptography is a traditional method of providing security and has made significant contributions in attack detections and preventions. However, due to the advancement in wireless communications, attackers have become more sophisticated and intelligent thus, making cryptography incapable of addressing all the attacks effectively. For these reasons, trust models have been proposed to supplement the cryptographic solution. In section 2.3, we review the basics of trust models and discuss several attacks that may affect the trust evaluation process of the trust model. We also provide a comprehensive review of existing trust-based schemes and discuss their limitations. Finally, in section 2.4, we present a survey of security works that have been proposed to overcome the drawbacks of deploying PKI. Based on this review, we hope to gain a better understanding of the requirements and shortcomings of existing works as a guide for designing more robust solutions.

2.1 Attacks in V-Mesh Network

V-Mesh networks are vulnerable to a wide range of attacks [5, 6, 39, 40, 41]. This can be attributed to unique features such as the open nature of the wireless medium, high mobility, dynamic topology, frequent disconnection and short connection times. While traditional wireless networks suffer from similar threats, attacks in V-Mesh networks are more dangerous due to the threats to human life. In this section, we present a survey of attacks in V-Mesh networks. We then elaborate on the security goals of communication and summarize the attacks that compromise these goals.

2.1.1 Classification of Attacks

Attacks on V-Mesh networks can be classified into two main categories namely, external attacks and internal attacks. External attacks are conducted by attackers who do not belong to the network. They are considered “outsiders" and cannot access any of the services provided by the network. The only way for these “outsiders" to launch an attack is to eavesdrop on the traffic or flood the network with fake messages. Thus, external attacks can be easily prevented using modern cryptographic techniques such as encryption, access control mechanisms, and firewalls.

Internal attacks are launched by authenticated users or “insiders" in the network. They have valid cryptographic keys and are familiar with the system policies and procedures. Since these attackers are part of the network, internal attacks are more severe and difficult to detect than external attacks. In addition, external and internal attacks can be classified into passive and active attacks.

Passive attacks are attacks that do not involve any data modifications but are intended to gain sensitive information or data through monitoring of the communication channel. In many cases, passive attacks are difficult to detect since the operation of the network is not affected. Active attacks, on the other hand, involve alteration, injection, replaying, and deletion of messages in the network. Due to this, the impact of active attacks on the network is more destructive than passive attacks in terms of scope and severity. Figure 2.1 shows a classification of different attacks in V-Mesh network and Table 2.1 provides a brief description of the various attacks listed in the figure. For a more detailed explanation, we refer the readers to the survey papers presented in [5, 6, 39, 40, 41].

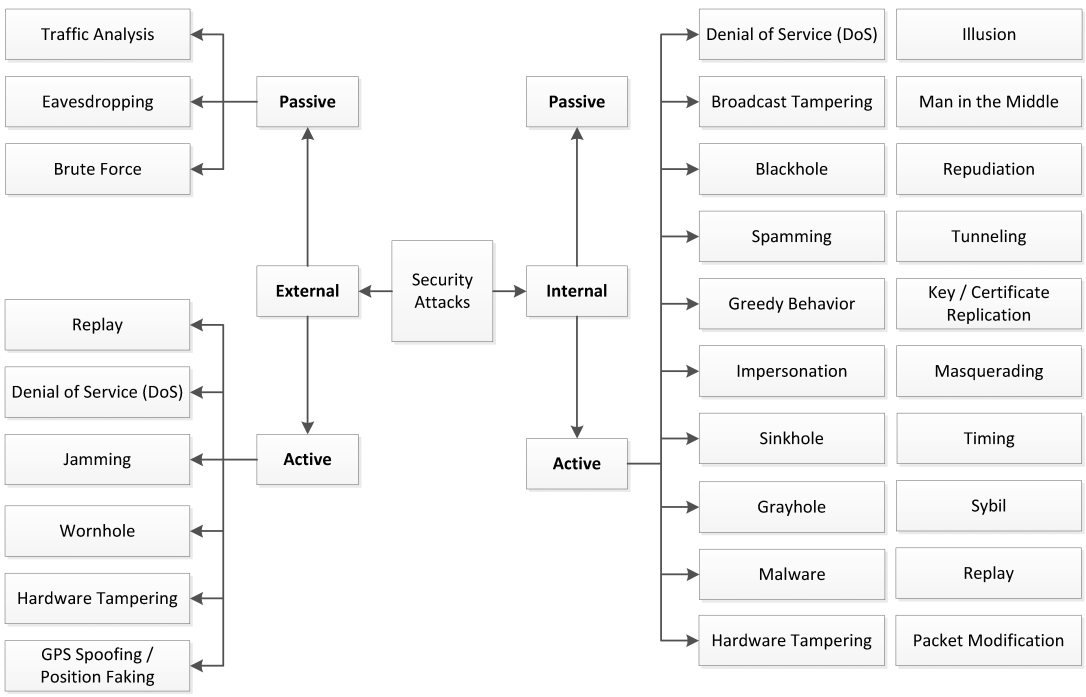

Figure 2.1: Classification of attacks.

Table 2.1: Description of attacks

Type of Attacks	Attack Descriptions
Traffic Analysis	Analyze collected information in an attempt to recover sensitive information.
Eavesdropping	Passively listen in on a message exchange to collect data but does not change the packets.
Brute Force	Trying to identify e.g. ID of a node through a exhaustive dictionary search process. Not easy as the connection times between two vehicles or RSUs is short
Replay	Saves a copy of the message and later uses it for replaying. Can be performed by external and internal attackers.
DoS	Flood dummy messages intentionally to overwhelm the receivers, thus disrupting services. Can be performed by external and internal attackers.
Jamming	Transmit a strong signal to jam the channel intentionally to disrupt or deny the usage of channel by legitimate users.
Wormhole	Establish a tunnel between two or more vehicles to bypass legitimate car users. Allows attackers to coordinate attacks from a distant. The tunnel is established using a radio channel different the network.

Continued on next page

Table 2.1 – *Continued from previous page*

Type of Attacks	**Attack Descriptions**
Hardware Tampering	Compromise a node physically e.g. HSM of vehicle or RSU. Can be performed by external and internal attackers.
GPS Spoofing / Position Faking	Use GPS satellite simulator to generate false location signals that are stronger than those generated by the actual satellite systems to deceive legitimate vehicles.
Broadcast Tampering	Inject fake security alert messages to hide the true safety messages to legitimate users.
Blackhole	Refuse to transmit messages received from legitimate vehicles by dropping the packets. Can be performed by external attackers and internal attackers.
Spamming	Sends useless messages such as advertisements to consume bandwidth and cause unnecessary congestion in the network including delay in processing other legitimate requests.
Greedy Behavior	Attacks the MAC layer by modifying the back-off algorithm so as to minimize the waiting time for faster access to the channel.
Impersonation	Usurp someone's identity to disguise itself as a legitimate user and sends message on behalf of other vehicles to create disruptions without disclosing itself.
Sinkhole	Different from blackhole in that the attacker attracts the neighbouring node so that their packets go through it. After that, the packets are either modified or dropped.
Grayhole	A variant of blackhole attack whereby a portion of the data packets are removed.
Malware	Injection of malicious software during software updates of vehicles to disrupt normal functioning.
Illusion	Place sensors that generate false data to create an illusion.
Man in the Middle	Listen to communication between two vehicles and modify or manipulate the information.
Repudiation	Deny having send or receive a message in case of dispute. Also mean denial of participation in all or part of the communication
Tunneling	Different from wormhole attack, the attacker use the same network to establish a private channel. Thus, this attack is performed by internal attacker.
Key/Certificate Replication	Use of duplicate keys or certificates as proof of identification to take part in a communication. This leads to ambiguity which makes tracking in the case of dispute difficult.

Continued on next page

Table 2.1 – *Continued from previous page*

Type of Attacks	**Attack Descriptions**
Masquerading	Fakes and pretends to be another vehicle through fabrication, alteration and replay. Eg. Pretend to be an ambulance.
Timing	Delay in forwarding important messages such as emergency and safety messages.
Sybil	Cheat with multiple false identities to simulate multiple nodes and transmits multiple messages to simulate there are many vehicles on the road.
Packet Modification	Modify, delete, construct or alter existing data to bring harm to other users or network.

2.1.2 Attacks on Security Goals

In addition to knowing the type of attacker and the nature of attack, it is also important to know the intent of the attacker or the security goals the attacker is trying to compromise. To this end, five security goals are defined in information security which we described below. We then classify the attacks discussed in the previous section on the basis of these security goals in Table 2.2.

- **Confidentiality** (C) refers to the protection of information from unauthorized access or disclosure. Only authorized nodes can access the information. When confidentiality is compromised, it usually means loss of privacy. As such, confidentiality and privacy are used interchangeable. Ensuring confidentiality and privacy can be achieved by means of encryption.

- **Integrity** (I) refers to the protection of information against unauthorized modification. Violation to the integrity property can have serious impact which can lead to catastrophe consequences such as loss of lives.

- **Availability** (A) refers to the protection of information from unauthorized disruption. This property requires that network services and information are always available to authorized users when needed.

- **Authentication** (AU) refers to the process of establishing that a user is a legitimate user in the network or if the users are who or what they claims to be. This property is crucial in a V-Mesh network since a vehicle or a pedestrian usually reacts upon the information received from others.

- **Non-repudiation** (NR) refers to the assurance that a sender cannot deny having sent a message. Similarly, the recipient cannot deny the receipt of message after it has been received. This property is required in the case of liability tracking.

Table 2.2: Attacks on security goals

Security Goals / Type of Attacks	C	I	A	AU	NR
Masquerading	✓	✓		✓	✓
Impersonation	✓	✓		✓	✓
Malware	✓	✓	✓		
Wormhole		✓	✓	✓	
Tunneling		✓	✓	✓	
Man in the Middle	✓	✓		✓	
Sinkhole		✓	✓		
Packet Modification		✓			✓
Sybil			✓	✓	
Brute Force	✓			✓	
Replay		✓		✓	
Hardware Tampering		✓	✓		
Illusion		✓		✓	
Key/Certificate Replication	✓			✓	
Blackhole			✓		
Grayhole			✓		
Traffic Analysis	✓				
Eavesdropping	✓				
DoS			✓		
Jamming		✓			
GPS Spoofing / Position Faking				✓	
Broadcast Tampering		✓			
Repudiation					✓
Timing			✓		
Spamming			✓		
Greedy Behavior			✓		

C: Confidentiality, I: Integrity, A: Availability, AU: Authentication, NR: Non-repudiation

2.2 Challenges in V-Mesh Network

Despite the huge application support for C-ITS applications, there are still many challenges that need to be addressed in V-Mesh networks. To begin our study, we surveyed about 100 papers related to WMNs [42, 43] and VANETs [2, 39, 6, 5, 31], including Mobile Ad-Hoc Networks (MANETs) [44] and divided the challenges in these networks into five main categories: key management, routing, access control, authentication and privacy [1]. Based on our survey, routing and key management are the two biggest challenges as shown in Figure 2.2. Routing is important because many vehicular applications and services rely on it for efficient data transfer over several hops. So naturally, routing is a potential attack surface in which an attacker can exploit to disrupt vehicular communications. On the other hand, designing a secure and efficient key management framework to create, distribute and protect the cryptographic keys for establishing secure communications is challenging because of the high mobility of the vehicles in the V-Mesh network. The frequent disconnections and short connection times may sometimes complicate the task of developing security solutions. For these reasons, the main contributions of this thesis are centered around the routing and key management issues to provide trusted and secure communications in a V-Mesh network.

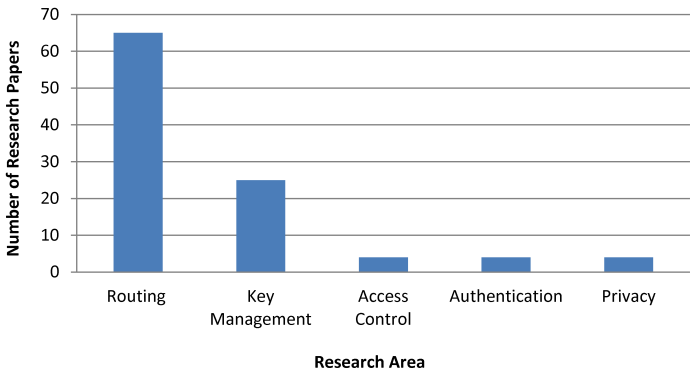

Figure 2.2: Classification of challenges

[1]Due to the lack of prior research studies on V-Mesh network, the security challenges presented in this thesis are based primarily on vulnerabilities in general wireless network. This thesis does not consider other security challenges caused by the interoperability between WMN and VANET. Nevertheless, we hope that our work will be the starting point for researchers to continue to explore and research new challenges in the future.

2.2.1 Routing Issues

Routing is an integral part of V-Mesh communications. All the different types of communication require routing to work properly. If a vehicle wants to access a particular service in the Internet, the request must go through the RSU and probably several hops in the mesh backbone before eventually reaching the gateway to connect to the Internet. Likewise, in a V2V communication, if the intended recipient is outside the communication range of the source vehicle, it is possible for the packet to travel several hops between different vehicles to arrive at the final destination. For these reasons, the routing protocol is very important for reliable data transfer. The purpose of the routing protocol is to determine the best route from the source to the destination to deliver the packets in the most time-efficient manner.

Unfortunately, most routing protocols are designed without security considerations. These routing protocols assume that the participating nodes are always trusted and are willing to cooperate with each other. In fact, some nodes can attack the routing protocol for selfish and malicious reasons. For example, a selfish node may refuse to forward the data packets in order to conserve scarce resources such as bandwidth and power. Another node may act maliciously by injecting false information or modify existing data packets during the forwarding process. Due to these misbehaviors, the network suffers from serious network degradation and disruption. To this end, we have identified some attacks that affect the routing based on our discussions in Table 2.1. They are the blackhole, grayhole, wormhole, tunneling, sinkhole and packet modification attacks. However, in this thesis, we only focus on blackhole, grayhole attacks, and packet modification attacks.

2.2.1.1 Blackhole Attack

A blackhole attack is an attack caused by the selfish behavior of nodes. In this attack, the selfish node exploits the routing protocol by advertising itself as having a valid route to the destination even though it does not have a route to it. This is achieved by sending a Route Reply (RREP) packet immediately to the source node. When the source node receives the first copy of the RREP packet, it assumes that the route discovery is complete and will ignore all other RREPs from the other

nodes. As a result, a path is selected that includes the selfish node as a forwarder to forward the data packets. Subsequently, the selfish node consumes all the traffic without forwarding it to the next hop and thus, create a blackhole in the network. As summarized in Table 2.2, blackhole attacks compromise the availability of the network services and can be considered a type of DoS attacks.

2.2.1.2 Grayhole Attack

A grayhole attack is a variant of the blackhole attacks, but is a more subtle form of dropping attack. Instead of dropping all the packets it receives, the attacker drops the packets with some probability. For this reason, the grayhole attacks are harder to detect because data loss may be perceived as a loss due to the network congestion, which is a natural phenomenon in all wireless networks. In some cases, the selfish node can choose to drop the traffic only from a particular node but continues to forward the packets for other nodes. A grayhole node may also inhibit on-off behavior where it appears to be well-behaved at one instant and selfish at the other. Consequently, the grayhole attacks may go undetected for a longer period of time than the blackhole attacks. The effect of grayhole attack is reduced throughput. Similar to the blackhole attack, grayhole attack affects the availability of services.

2.2.1.3 Packet Modification Attack

In this attack, an attacker modifies a specific part or all of the message to create confusion and harm the reliability of C-ITS applications. For example, a malicious attacker can tamper with a received packet to indicate road congestion so that other vehicles can avoid a particular area. An attacker can also alter the contents of the packet to claim that there is no accident at the road intersection. If this information is propagated throughout the network, other vehicles may be caught in unnecessary traffic jams which result in loss of time and money. In the worst case, collisions can happen in the same area which makes recovery even more challenging. Unlike blackhole and grayhole attacks, packet modification violates the integrity of the received data. It also violates the non-repudiation property if certain part of the message which is used for tracing purposes is erased deliberately.

2.2.2 Key Management Issues

Key management is another key aspect of V-Mesh deployment because it is responsible for the management of cryptographic keys in a cryptosystem. This involves generation, distribution, storage, and revocation of keys between nodes in the network. As a guideline, the IEEE 1609.2 [29] and the ETSI TS 102 940 standards [33] have proposed the use of PKI [45] to manage the keys and provide communication security for C-ITS applications. In the following, we present a traditional PKI architecture and analyze the shortcomings of using PKI in a V-Mesh network.

2.2.2.1 Traditional PKI

PKI uses asymmetric cryptography and digital certificates to provide confidentiality, authenticity, integrity, and non-repudiation. These certificates are used to prove whether the security credentials belong to a certain node in a communication.

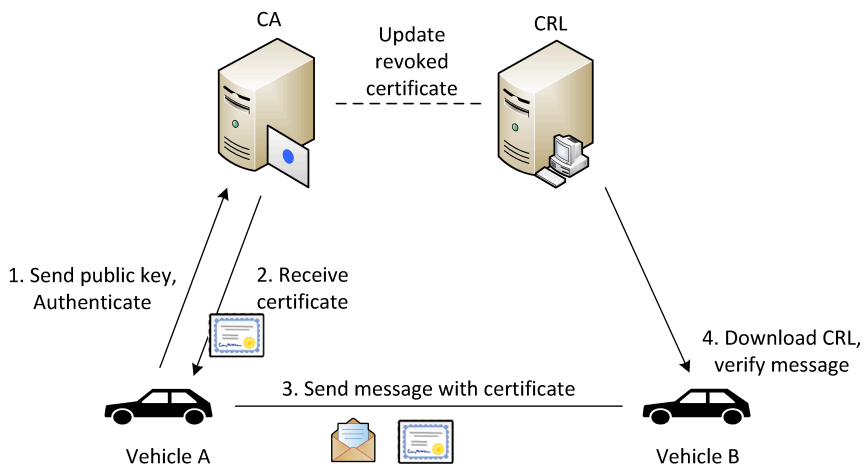

Figure 2.3: Traditional PKI architecture.

With reference to Figure 2.3, the steps involved in the PKI operation can be summarized as follows:

1. The vehicle A requests a certificate from the CA by authenticating itself using valid credentials.

2. The CA verifies that vehicle A's credentials, i.e. identity is valid and is not revoked. After vehicle A is authenticated, CA issues the issued certificate to

vehicle A.

3. Once vehicle A has obtained the certificate, it can start a communication with vehicle B. Basically, vehicle A signs all outgoing messages using its private key and attaches the issued certificate to the message to vehicle B.

4. When vehicle B receives the message from vehicle A, it verifies vehicle A's certificate first before verifying the vehicle A's message. This entails downloading the CRL from the RSU in the vicinity. If the certificate is valid and is not revoked, vehicle B retrieves the public key of vehicle A in order to verify the message and the process stops here.

Note that if vehicle A is to be revoked, the CA will update this information into the CRL which will then be disseminated during the next window.

2.2.2.2 Issues of PKI

The main problem with PKI is that the sender must attach his/her digital certificate to the message to prove ownership of the keys. This results in high message overhead and wasted communication bandwidth. Second, when a certificate is received, the recipient must:

1. Check the expiry date of the certificate,

2. Download the CRL,

3. Verify the validity of the certificate against the CRL,

4. Verify the CA's signature on the received signed certificate,

5. Retrieve sender's public key and verify sender's message.

This results in a huge latency which cannot meet the requirements of safety-related applications [46]. As an example, the maximum latency for disseminating safety messages in a forward collision application is 100 ms [2, 30]. If the latency is high, critical messages will not reach the other users in time, which can have catastrophic consequences.

Moreover, in the case of a high certificate revocation rate, the size of the CRL will increase significantly. This further increases the latency of transmitting and

downloading the CRL. In addition, CRL is only published on an hourly, daily or weekly basis. There is an issue with the freshness of the key where the recipient uses a certificate that has been revoked. [45]. One solution is to disseminate the CRL more frequently. However, this leads to an increase in the communication cost.

Finally, PKI uses asymmetric cryptography, which is designed based on a trapdoor function that is easy to compute in one direction but is difficult to reverse the operation. This requires that the public key and the private key are large numbers so that the private key is not easily derived from the public key. Even if Elliptic Curve Cryptography (ECC) based algorithms are employed, the encryption, generation and verification of signatures can be slow [47].

2.3 Review of Trust-based Routing

There are generally two methods to protect the routing against routing attacks and can be classified into two types, namely cryptographic-based routing and trust-based routing.

Cryptographic-based routing relies on the security of cryptography to protect the routing path. Schemes that employ cryptography use a variety of cryptographic tools such as encryption, hashing, and digital signatures to prevent information disclosure, message modification and impersonation attacks. However, these cryptographic-based routing schemes are incapable of tracking attacks that are launched by internal nodes. More specifically, they cannot detect packet dropping attacks such as blackhole and grayhole attacks. Moreover, if the nodes are physically compromised, the attackers can gain access to the cryptographic keys and breach the system. For these reasons, trust models have been proposed and incorporated into the routing protocols as a second line of defense to complement the cryptography-based routing. In this thesis, we focus on trust-based routing and refer the readers to [6, 48, 49] for more details on cryptographic-based routing.

2.3.1 Components of Trust Model

Trust-based routing, as the name implies, relies on the trust model and the concept of trust to protect the route. The trust model is used to evaluate the behavior of other nodes and build trust values based on the observed actions. Trust

values are communicated between other nodes in the network to help them establish trust relationships. Each node may then use the aggregated trust values to determine the trustworthiness of other nodes, make choices as to which nodes to cooperate, and even take action to punish an untrustworthy node if necessary. The whole process can be described using three components, namely the trust monitoring, the trust aggregation and decision making as shown in Figure 2.4. In the following, we elaborate on these components and discuss the shortcomings (if any) associated with these components.

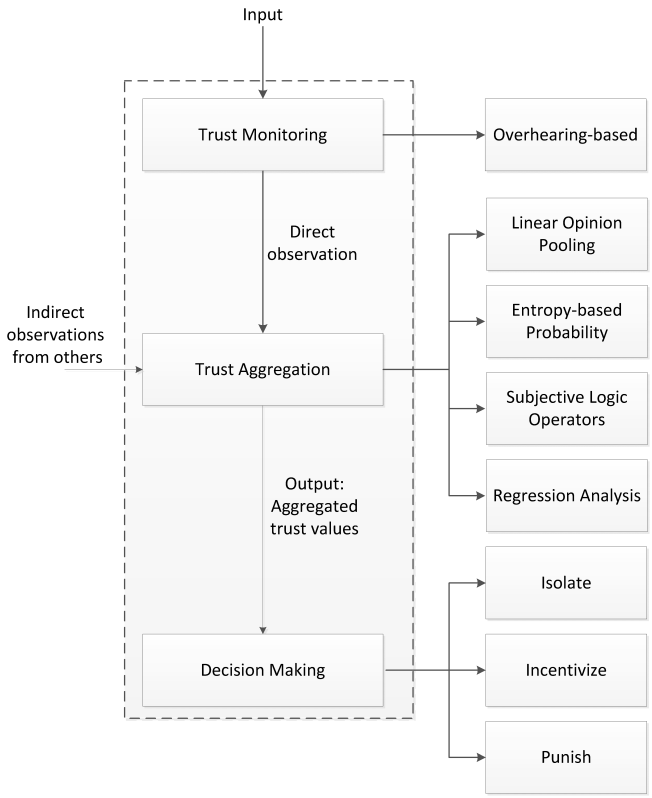

Figure 2.4: Components of trust model.

2.3.1.1 Trust Monitoring

The trust monitoring component monitors the behavior of a node and outputs the trust value as direct observations. In order to monitor the behavior of nodes, the most well-known mechanism, as shown in Figure 2.4, was proposed by Marti et al. in 2000 [36] and is known as the overhearing technique. The basic idea is to exploit

the broadcast nature of the wireless medium to overhear the transmission of the downstream node. By overhearing, the upstream node can determine whether the downstream node has forwarded the packets it receives. If the number of dropped packets exceeds a certain threshold, the downstream node is deemed as misbehaving. Due to its simplicity, many researchers have adopted this technique as their monitoring tool to detect blackhole and grayhole attacks. However, the drawback of overhearing is that the overhearing node may not able to overhear every transmission due to the inevitable channel fading, collisions with other transmissions and interference [36]. In addition to the unreliable wireless channel, a malicious node may misbehave to obstruct the overhearing process. In the following, we discuss two attacks that can impact the overhearing process.

- **Limited transmission power attack**: In this attack, an attacker controls its transmission power so that the packet transmission is overheard by the previous hop, but is too weak to be received by the next hop. Consequently, the previous node always detects the attacker as a trusted node and include the attacker as part of the routing path. This distorts the overhearing process and produces incorrect judgments that undermines the accuracy of the trust evaluation. The impact of such attack is packet loss or reduced network throughput.

- **Packet modification attack**: In this attack, an attacker modifies the message before forwarding it and still get a positive rating from the trust evaluation process. As a result, the packet modification attackers will always be evaluated as a trusted node to perform packet forwarding. The effects of such attack can be detrimental to the security of C-ITS applications. Therefore, appropriate measures must be taken to mitigate these attacks.

2.3.1.2 Trust Aggregation

Trust values are usually calculated based on direct observation and indirect observations. Here, the direct observation refers to the output of the trust monitoring component, wherein the trust values are calculated based on personal experiences. On the other hand, the indirect observation refers to the trust values collected from other nodes in the network. Therefore, the function of the trust aggregation compo-

nent is to combine the indirect and direct observations together. To this end, several trust aggregation techniques have been proposed which will be discussed later. The aggregation step aims to improve the accuracy of trust evaluations and accelerate the detection time of selfish nodes in the network [50, 51, 52, 53]. However, as described in [54, 55, 56, 57, 58], a malicious node can provide dishonest recommendations to subvert the trust evaluations. In the following, we discuss two attacks that can influence the trust aggregation process.

- **Badmouthing attack**: In this attack, the node serving as a recommender attempts to ruin the trust of a good node intentionally by providing bad recommendations against it so as to decrease the chance of that node being selected for service. The aim of badmouthing is to cause the well-behaved node to be blacklisted, thus, causing a partition in the network to decrease the throughput and the overall reliability of the network.

- **Ballot-stuffing attack**: In contrast to badmouthing attack, the malicious recommender attempts to boost the trust of another malicious node by providing good recommendations so as to increase the chance of that malicious node being selected as a forwarder. Subsequently, the trustee may conduct other packet dropping attacks such as blackhole or grayhole attacks. This form of attack can also be referred to as a form of collusion attacks where the malicious recommender and the trustee act in collusion.

2.3.1.3 Decision Making

The decision component uses the aggregated trust value as input and typically determines whether the node is trustworthy based on a detection threshold. In doing so, the trust model can distinguish betwen selfish nodes and well-behaved thereby assisting the routing protocol to achieve a secure routing path for forwarding packets. According to the type of trust models proposed, the actions taken by the decision component can be divided into isolation, incentive and punishment types, as shown in Figure 2.4. The difference between the first case and the last two cases is that a selfish node is isolated permanently upon detection while the incentive and punishment mechanisms are designed to facilitate the participation of the node over time.

2.3.2 Survey of Trust-based Schemes

In this section, we provide a review of several trust schemes proposed for different networks to motivate our research on the deficiencies of overhearing. These schemes can be classified into reputation-based approach and credit-based approach as shown in Figure 2.5.

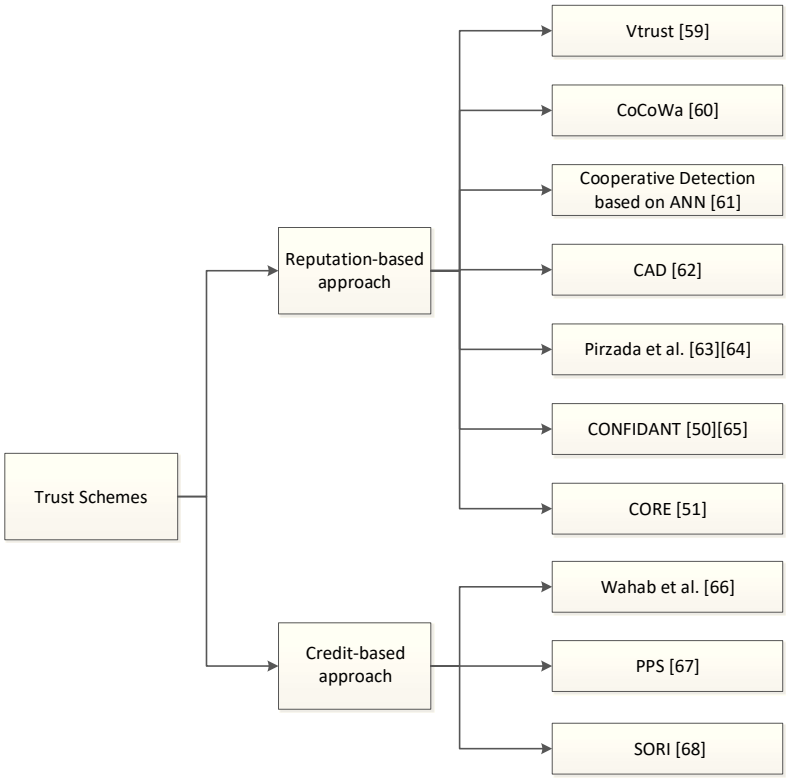

Figure 2.5: Classification of trust-based schemes

In the reputation-based approach, a monitoring process takes place to detect the presence of misbehaving nodes. The detection results are then disseminated across the network to prevent the use of misbehaved nodes in future routing. Basically, this approach adopts the monitor, detect, and isolate work flow to manage all misbehaved nodes. In the credit-based approach, the nodes are given motivations to perform networking functions. Thus, the work flow is monitoring, detection followed by either a punishment or incentive mechanism. If the node is well-behaved, credits in the form of payment are given to improve the node's reputation in the network. On the other hand, if the node misbehaves, negative payments are given to penalize

the node. If the credit or reputation runs out, the misbehaved node will not be able to send any packets anymore. In the following, the main contributions in both reputation-based and credit-based approaches are discussed.

2.3.2.1 Reputation-based Approach

Hu Hao et al. have proposed a trust framework called VTrust [59] to distinguish grayhole vehicles from well-behaved vehicles so that vehicles with high trust score are always selected as relays, thereby increasing the efficiency and reliability of VANET. Besides measuring the forwarding behavior of the nodes, the social similarities of the vehicles and the messages are also considered in the trust evaluation since a vehicle with more social similarities with the message is more likely to forward the message. By incorporating social attributes, it ensures that the best relay is always selected for forwarding which improve the robustness of data dissemination in vehicular communications.

E. Hernandez-Orallo et al. have proposed a contact-based scheme called Collaborative Contact-Based Watchdog (CoCoWa) to improve the dissemination of selfish node information [60]. The information is disseminated based on contacts rather than flooding. In doing so, it reduces the detection time of selfish nodes and reduces message overhead compared to the traditional overhearing. Furthermore, to overcome the effects of false positives, i.e. the non-selfish nodes are identified as selfish nodes, the authors introduce a negative diffusion factor in which negative detections from other nodes are transmitted to counteract their effects. Here, the negative detections mean that a node is not selfish. However, the negative diffusion factor must be properly tuned to achieve the desired behavior.

A. EL. Khatib et al. have proposed a cooperative detection mechanism [61] based on the Artificial Neural Network (ANN) to detect misbehaving nodes. Their scheme makes use of overhearing to track the nodes' behaviors. After that, ANN is used to aggregate the observations from multiple sources. The use of ANN is twofold. First, it overcomes the limitation of DST when observations come from dependent sources. Second, the detection mechanism can benefit from the previous detection experience by employing continuous learning.

Shila et al. have proposed a Channel Aware Detection (CAD) algorithm to distinguish grayhole attacks from normal losses [62]. Their approach is based on two

procedures, namely traffic monitoring and channel estimation. In the traffic monitoring process, the behavior of a node is determined by downstream and upstream monitoring. As shown in Figure 2.6, node A performs downstream monitoring on node B to determine whether node B is dropping or tampering the data packets while node C performs upstream monitoring on node B to measure the loss rate of the link between node B and node C. In the channel estimation process, the normal loss rate due to the poor channel quality and medium access collision is estimated to determine the downstream and upstream detection thresholds. The traffic monitoring observations are then compared to the downstream/upstream detection thresholds to detect misbehaviors. By using downstream and upstream monitoring, the CAD scheme can also detect limited transmission power attacks. And to detect packet modification attacks, the CAD scheme relies on overhearing to verify the integrity of each forwarded packets.

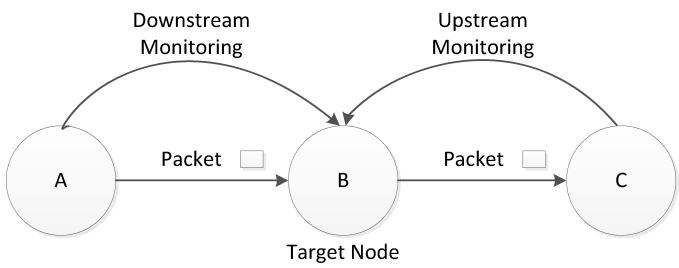

Figure 2.6: Traffic monitoring procedure

Pirzada and McDonald have proposed to classify the multiple observed events in the network into different trust categories [63] such as the acknowledgment, packet precision, gratuitous route replies, blacklists, salvaging etc. For each trust category, the information gathered is based on the overhearing technique. The final trust is then computed by aggregating all the different trust categories according to their assigned weights. This work is further extended to other reactive routing protocols in [64] where a comparative study is presented to study the effectiveness of using trust categories. The authors have considered the packet modification attacks in which the proposed solution is to buffer the packets for a specified time interval and use overhearing to verify the integrity of the forwarded packets.

Other notable trust schemes include the Cooperation of Nodes: Fairness in Dynamic Ad-hoc NeTworks (CONFIDANT) protocol proposed by S. Buchegger et al.

[50, 65]. It is a framework consisting of four components: a monitor, a trust manager, a reputation manager and a path manager to detect and punish selfish nodes. The monitor component is in charge of monitoring the communication between the neighboring nodes promiscuously. The trust manager component is responsible for triggering alarm messages when misbehavior is detected. The reputation manager gathers the alarm messages from other observing nodes and updates the node rating. If the node rating is unacceptable, the reputation manager triggers the path manager which deletes the misbehaving nodes from all the active routes in the path cache. In [51], P. Michiardi and R. Molva have proposed the COllaborative REputation (CORE) mechanism that employs a more elaborate trust model to enforce cooperation among nodes. In particular, trust in their scheme consists of subjective reputation, indirect reputation, and functional reputation. In contrast to the CONFIDANT protocol, only positive observations are disseminated in CORE to prevent DoS attacks towards well-behaved nodes.

2.3.2.2 Credit-based Approach

O. A. Wahab et al. have proposed a cooperative overhearing-based clustering scheme for VANETs [66] to enforce node cooperation during clusters' formation and detect misbehaving vehicles after cluster formation. In their scheme, cluster head and Multipoint Relay (MPR) nodes are selected based on Quality of Service (QoS) metric calculated from the bandwidth, connectivity, velocity, and the residual distance. In order to motivate cooperation during cluster formation, payments are rewarded to the elected cluster heads and the MPR nodes. After the election, the cluster members and cluster heads begin to monitor the behavior of the MPR nodes using overhearing. The observations are shared among all the nodes in the same cluster so that each node can aggregate the observations using the DST technique to make the final decision. If the MPR behaves normally, payment is issued to facilitate further cooperation. If the MPR node is identified as a selfish node, it will be blacklisted and other nodes will refrain from electing or cooperating with it in future.

Jesudoss et al. have proposed a Payment Punishment Scheme (PPS) scheme to encourage nodes to cooperate during cluster formation and packet forwarding [67]. The idea is to reward incentives or payments to the nodes to motivate them

to perform these functions. The payment is designed based on the Vickrey-Clarke-Groves (VCG) model, which uses game theory to make cooperative behavior the dominant strategy of all participants. To reward the nodes for packet forwarding, each node uses overhearing to monitor the forwarding behavior of the relay node and sends the trust report to the cluster head. The cluster head, then collects at least three trust reports from the other nodes and aggregates them using the extended DST combination rule. If the relay node is cooperative in nature, positive payment is granted to the relay node. Otherwise, a negative payment is issued to reduce its reputation in the network.

Q. He et al. have proposed the Secure and Objective Reputation-based Incentive (SORI) scheme which consists of three phases, namely neighbor monitoring, reputation propagation, and punishment [68]. Neighbor monitoring uses overhearing to gather information about the packet forwarding behavior of the nodes in order to compute the trust value. Reputation propagation shares information among neighboring nodes to make the reputation measure more accurate. In order to secure the propagation of reputation, a one-way hash chain is implemented to prevent impersonation attacks. The punishment phase is responsible for penalizing selfish nodes by probabilistically dropping the packets that originate from the selfish node. By punishing the misbehaved nodes, SORI instills cooperation and improve packet forwarding in the network.

2.3.2.3 Discussions

Based on our review, all the schemes above rely on overhearing to monitor the forwarding behavior of the nodes. As a result, they are vulnerable to limited transmission power attacks and packet modification attacks. Moreover, the accuracy of overhearing is limited by the unreliable wireless channel such as ambiguous collisions and channel quality [36] that may hinder the monitoring process.

To solve the inaccuracies of detection, schemes like [66, 67] have proposed the use of DST to aggregate the observations. However, they are only effective if evidences from the other vehicles are available. If the network is sparse such that the decision is based on the local observations of a single watchdog, these schemes still suffer from the imperfections of overhearing.

In terms of packet modification attacks, [67], [63] and [62] have proposed that

each node checks the integrity of the forwarded packet when it overhears the next hop transmission. In particular, [67] have proposed that each node generates a hash of the data packet, and store it before propagating to the next hop. When the data packet is forwarded by the next hop, it computes the hash of the forwarded packet, and compares it with the stored hash value. If the two hash values match, it means that the data is consistent. Despite these measures, overhearing fails when the wireless channel condition is bad.

For limited power transmission attacks, the CAD algorithm [62] is by far the only solution to this attack. The reason is that the CAD algorithm uses downstream and upstream monitoring to measure the node behavior. If an attacker performs a limited transmission power attack, the attack will be detected by upstream monitoring process because the observed loss rate will exceed the upstream detection threshold. However, the downside of CAD is that the detection process is slow because the source node needs to send multiple query packets to locate the malicious node hop by hop. In addition, the CAD scheme does not rely on indirect trusts which have been shown in [50, 51, 52, 53] to improve the detection time of misbehaved nodes. Given that the overhearing mechanism is the cornerstone of many trust models proposed to date, it is important that these shortcomings are properly rectified such that appropriate measures can be incorporated into the future trust models to enhance their robustness.

2.3.3 Survey of Trust Aggregation Techniques

Several trust aggregation techniques have been proposed to combine direct and indirect trust values. In this section, we review four different trust aggregation techniques, which can be classified into linear opinion pooling, entropy-based probability model, subjective logic and regression analysis.

2.3.3.1 Linear Opinion Pooling

A traditional method of aggregating trust opinions is by the linear opinion pooling method [69] proposed in [53, 70, 57, 71, 72]. In this approach, the indirect trust is weighted first before it is combined with the direct trust as shown in (2.1).

$$T_{i,j}^{aggregated} = T_{i,j} + \omega T_{k,j} \tag{2.1}$$

where $T_{i,j}$ denotes the direct trust of node j as observed by node i, $T_{k,j}$ denotes the indirect trust of node j as observed by recommender k, $T_{i,j}^{aggregated}$ denotes the aggregated trust of node j as derived by node i and ω is a weighting factor to adjust the effects of indirect trusts.

In general, ω is selected to be any arbitrary value in the range (0,1) as proposed by [53, 70]. In a more specific case, authors in [57, 71, 72] have proposed to use the trust level of the recommender to weigh the indirect trusts such that indirect trust that originates from an untrustworthy recommender are discounted more than the one that comes from a highly reputed recommender. A slight difference between [71] and [57, 72] though, is that the direct trust component in [71] is also weighted using the double weighted approach where the sum of the weights given to the direct and indirect trust components does not exceed 1.0.

2.3.3.2 Entropy-based Probability Model

In [54, 56, 58, 55], Y. Sun et al. have proposed a probability-based model to characterize trust propagation through a recommender. Suppose node i wants to establish the trust level of node j through a recommender k, the trust concatenation model defines the formula in (2.2) to merge the trust probabilities.

$$P_{i,j} = P_{i,k}P_{k,j} + (1 - P_{i,k})(1 - P_{k,j}) \tag{2.2}$$

where $P_{i,j}$ is the probability that node j carries out an action, $P_{i,k}$ is the probability that recommender k makes good recommendations and $P_{k,j}$ is the probability that node j carries out some actions in the recommender's view. In this case, the probability values are estimated by evaluating the expected value of the beta distribution [73].

When there are multiple paths between node i and node j, the expected value and the variance of the each path m are first converted to beta parameters (α_m, β_m) before they are combined together using (2.3).

$$\begin{aligned} \alpha &= \sum_{i=0}^{m} \alpha_i \\ \beta &= \sum_{i=0}^{m} \beta_i \\ \forall i &= 1, \cdots, m \end{aligned} \tag{2.3}$$

After that, the aggregated probability can be found by evaluating the expected value of the beta distribution following which the aggregated probability is converted to trust values using the entropy function as described in (2.4) and (2.5).

$$T_{i,j} = \begin{cases} 1 - H(P_{i,j}), & \text{for } 0.5 \leq P_{i,j} \leq 1 \\ H(P_{i,j}) - 1, & \text{for } 0 \leq P_{i,j} < 0.5 \end{cases} \tag{2.4}$$

$$H(P_{i,j}) = -P_{i,j} log_2(P_{i,j}) - (1 - P_{i,j}) log_2(1 - P_{i,j}) \tag{2.5}$$

where $H(P_{i,j})$ is the binary entropy function, $T_{i,j}$ denotes the trust of node j as perceived by node i and $P_{i,j}$ denotes the aggregated probability value.

2.3.3.3 Subjective Logic Operators

In [74], Josang have proposed an algebra to quantify uncertainty in trust relationships between entities. Such uncertainties occur due to the imperfect knowledge about the reality or due to the lack of evidence. Consequently, this leads to the notions of belief, disbelief and uncertainty which forms the basis of subjective logic.

In this model, trust is treated as a subjective opinion consisting of four parameters $(b_{i,j}, d_{i,j}, u_{i,j}, a_{i,j})$ where $b_{i,j}$ and $d_{i,j}$ represents node i's belief and disbelief in node j, $u_{i,j}$ represents the amount of uncertainties regarding the observations and $a_{i,j}$ is the a prior probability in the absence of evidence. These parameters have the following relationships and satisfy the following conditions in (2.6).

$$b_{i,j} + d_{i,j} + u_{i,j} = 1.0, \quad b_{i,j}, d_{i,j}, u_{i,j}, a_{i,j} \in [0.0, 1.0] \tag{2.6}$$

From the definition of an opinion, the trustworthiness of a node is derived by evaluating the expectation of the subjective opinion as defined in (2.7).

$$T_{i,j} = b_{i,j} + a_{i,j} u_{i,j} \tag{2.7}$$

where $T_{i,j}$ represents node i's trust in j and the parameter $a_{i,j}$ determines how much uncertainty contributes towards the trust computation. Since subjective logic is able to account for uncertainty, it improves the clarity and expressiveness of an opinion compared to the traditional probabilistic logic. The subjective logic further establishes two operations [75, 76] to handle trust propagations in a network.

The first operator is called the discounting operator which is used to derive a trust opinion from transitive paths via a recommender. The second operator is called the consensus operator and is used to derive a trust opinion from multiple parallel paths. The concept of subjective logic has been very popular and widely used in MANETs [52, 77, 78] and WMNs [79] including VANETs [80] with some slight variations.

In [77], the trust opinions from multiple recommenders are first combined into a single opinion using the weighted average approach before it is fused with the direct trust using the consensus operator. In [78], the weighting factor for each recommendation trust opinion is derived based on a familiarity value which denotes the familiarity degree of the recommender with the target node to be evaluated. In [79], each of the recommendation trusts are weighted by the trustworthiness of the recommender. Subsequently, the recommendation trusts are aggregated using a simple average function. In [80], subjective logic is proposed as a tool to merge opinions coming from different misbehavior detection mechanisms together to enhance the trust accuracy and detection capability of the system.

2.3.3.4 Regression Analysis

In [81, 82], Y. Wang et al. have proposed the LogitTrust model which is based on regression analysis to predict the trustworthiness of nodes. It is a statistical process for understanding the relationships between the observations gathered by a node and the corresponding operational and environmental factors such as energy sensitivity, capability limitation and cost awareness. Trust, in this model, is expressed as a logistics function and is given by (2.8).

$$T_j^{t+1} = \frac{1}{1 + e^{-(x^t)^\top \beta_j}} \tag{2.8}$$

where T_j^{t+1} represents the trust of node j at time $t+1$, $(x^t)^\top$ is a column vector of variables that characterize the operational and environmental factors at time t and β_j is a vector of regression coefficients. The Expectation-Maximization (EM) algorithm is then used to estimate the regression coefficients such that the log-odds of a node in providing a satisfactory service is greater than 0. After that, the regression coefficients are substituted back into (2.8) to compute the trust of a node.

2.3.3.5 Discussions

As noted in [54, 55, 56, 57, 58], a malicious node can provide dishonest recommendations to badmouth honest nodes or ballot-stuff malicious nodes as long as recommendation trusts are taken into consideration. In order to deal with such attacks, additional constraints and further processing of the recommendations are often required during the trust aggregation process.

More specifically, [53, 70, 71, 72, 58] have proposed a threshold filtering mechanism to filter out the bad recommendations based on checking the trust level of the recommender as well as the deviation between the locally derived direct trusts and the received indirect trusts. If the recommender's trust level is above a pre-defined threshold and the deviation is within an acceptable threshold, the recommendations will be taken into consideration; otherwise, they are discarded.

Other schemes such as [51, 57, 56, 55] prevent badmouthing and ballot-stuffing attacks by imposing strict constraints such that only positive recommendations are allowed to propagate in the network while the rest of the schemes [54, 52, 77, 78, 79, 80] consider transitive trust whereby the received recommendations are weighted by the trust level of the party that provides the recommendation. On the other hand, the regression approach as proposed in [82] achieves resiliency by replacing the logistics distribution with a white noise in t-distribution. Since the t-distribution has heavier tails, records with a high variance are weighted down and therefore, producing a more accurate estimate. Table 2.3 summarizes the various schemes by means of aggregation techniques and the manner in which the recommendations are processed during the evaluation.

Table 2.3: Summary of trust-based schemes and aggregation techniques

Aggregation Technique	**Processing of Recommendations**	**Schemes**
Linear Opinion Pooling	Threshold-based filtering	• S. Buchegger et al. [53, 70], • R. Chen et al. [71], • R. Li et al. [72]
	Positive Recommendations	• S. Ganeriwal et al. [57]
Entropy-based Probability Model	Threshold-based filtering	• Y. Sun et al. [58]
	Positive Recommendations	• Y. Sun et al. [56, 55]
	Transitive Trust	• Y. Sun et al. [54]
Subjective Logic	Transitive Trust	• X. Li et al. [52],

Continued on next page

Table 2.3 – *Continued from previous page*

Aggregation Technique	**Processing of Recommendations**	**Schemes**
		• K. Kane et al. [77], • Y. Liu et al. [78], • H. Lin et al. [79], • S. Dietzel et al. [80]
Regression Analysis	Robust statistical kernel to tolerate outlier recommendations	• Y. Wang et al [82]

Despite these efforts, there are drawbacks to each approach. In the threshold-based filtering approach, it is challenging to determine the optimal threshold settings. If the threshold is set too high, many recommendation trusts will be excluded, which defeats the purpose of collecting recommendations. If the threshold is too low, false recommendations will be left to propagate in the network, which could affect the accuracy of the trust evaluations. Moreover, a malicious node may behave to be a trustworthy recommender to bypass the threshold check and then issues false recommendations to ruin the reputation of other nodes. In the worst case, the malicious recommender may misreport the recommendations randomly to only a certain group of nodes to undermine the effectiveness of the filtering mechanism. To avoid the setting of a threshold, Z. Ullah et al. have proposed a method [83] that is based on the median-based deviation and the smoothing factor concept presented in [84] to filter the dishonest recommendations. Although this approach does not use any threshold, it may not be suitable for use in situations when there are few recommendations. On the other hand, if there are many recommendations to process, the time-complexity will increase accordingly.

On the other hand, by allowing only good recommendations to propagate in the network, it limits the practicality of the trust systems and making it less adaptable to changes in the network. The weighting of the recommendations may not help since the malicious node can behave trustworthy in the eyes of the evaluator such that discounting has no effects on the final weighted trusts. While regression approach seems appropriate, this approach requires more samples to achieve a stable and accurate results, otherwise, it is highly sensitive to sampling bias. For these reasons, these schemes are considered biased, which inspire us to explore new ways to handle

recommendations in trust management.

2.4 Review of Key Management

In this section, we provide a review of security solutions aimed at reducing the high costs of deploying a PKI. These security solutions can be broadly classified into symmetric cryptography and asymmetric cryptography as shown in Figure 2.7. In the following, we further elaborate on the main contributions in these two areas.

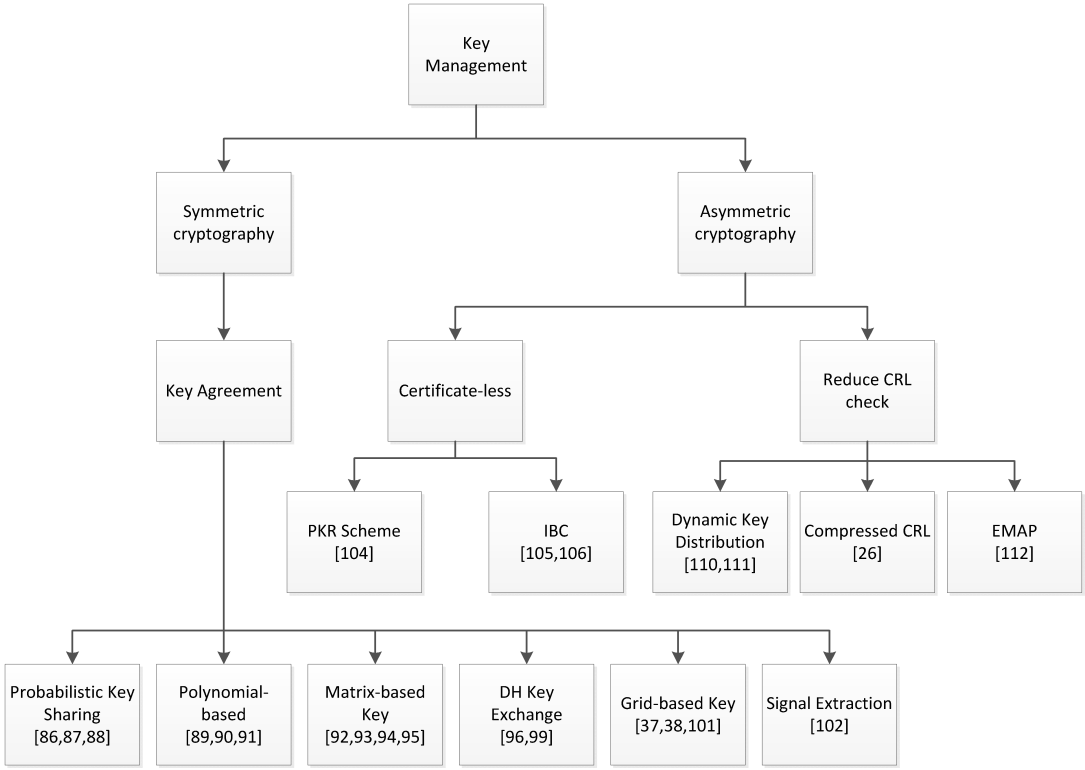

Figure 2.7: Classification of key management solutions

2.4.1 Symmetric Cryptography

The motivation for using symmetric cryptography to secure vehicular communications is that it is easier to implement and generally requires less processing power than asymmetric cryptography. This is due to the fact that the same key is used for encryption and decryption of data. By using symmetric cryptography, certificate management problems can be avoided, thus facilitating the implementation of

time-critical C-ITS applications. However, the main challenge of symmetric cryptography is that both parties involved must agree on the key in advance. If the keys are delivered over the air as the medium to the other party, this would be prone to security risks such as eavesdropping or being intercepted. For these reasons, several key distribution schemes have been proposed in the literature to exchange or establish the keys securely.

2.4.1.1 Key Distribution Schemes

In [85], the authors have proposed to use symmetric cryptography in conjunction with asymmetric cryptography within a PKI to reduce the complexity of certificate verification and to tackle the low performance issues of digitally signing all messages. In their proposal, the symmetric cryptography is used to secure telematics messages which are sent periodically whereas asymmetric cryptography is used to secure safety related broadcast messages which are not transmitted too frequently. However, details of how to generate the symmetric keys and distribute them is not discussed.

In [86, 87, 88], the authors have proposed a probabilistic key sharing scheme where each node is given a key ring containing k keys drawn randomly from a large pool of P keys. For any two nodes to determine a common key, each of them has to broadcast the list of the key identifiers on their key ring using a challenge-response protocol. However, the drawback is that the key storage requirement per node is high to ensure a high probability of sharing a key.

In the polynomial-based approach [89, 90, 91], the authority first generates a symmetric, bivariate polynomial $f(x, y)$ of degree t. Each node in the network is then preloaded with a univariate polynomial share of the $f(x, y)$ by substituting the assigned ID given to a node. For example, node u where u is the assigned ID, is given the univariate polynomial, $f(u, y)$. For any two nodes u and v to establish a shared key, node u can evaluate $f(u, v)$ whereas node v evaluates $f(v, u)$ at $y = u$. Since the univariate polynomials are extracted from a symmetric bivariate polynomial, it follows that $f(u, v) = f(v, u)$ and the two communicating nodes can derive the same common keys. The disadvantage of this approach is that it can only tolerate not more than t compromised nodes and increasing the t parameter increases the storage overhead.

In another approach called the matrix-based key establishment protocol [92, 93,

94, 95], a shared key can be established via the symmetric properties of the matrix operations $K = A \cdot G = (D \cdot G)^T \cdot G = G^T \cdot D \cdot G = (A \cdot G)^T$ where K is the symmetric key matrix containing the derived key elements. In this scheme, G is a public matrix considered as a public information known by all nodes in the network and is generated by the authority. D is a symmetrical secret matrix generated by the authority and should not be disclosed to the nodes in the network. A is a matrix which is the transpose of $D \cdot G$ and is made public to the nodes in the network. To derive a common key between node i and node j, node i multiplies the jth row of matrix A with the ith column of matrix G. Likewise node j multiplies the ith row of matrix A with j^{th} column of matrix G and it can be shown that k_{ji} as calculated by node i is equal to k_{ij} calculated by node j.

In [96], HT. Wu et al. have proposed the Diffie-Hellman (DH) key exchange [97, 98] and Hashed Message Authentication Code (HMAC) to provide message authentication in inter RSU, intra RSU communication range and hand-off among different RSUs. All message authentications are carried out by the RSU. In this method, DH key exchange protocol is used to establish a common secret key between a vehicle and a RSU. After that, a hash chain takes the DH secret key as an input and generates a number of session keys equal to the number of RSUs in the region. These computed session keys between the vehicle and the other RSUs are then communicated to the neighboring RSUs throughout the network. Any vehicle traveling in intra or inter RSU can then use the corresponding session keys to encrypt, decrypt and authenticate HMAC. However, the drawback of this method is that in inter RSU communication, the vehicle has to re-send messages in combination with the session key that is associated with the corresponding RSU in order for vehicles under different RSU's communication range to authenticate which increases the transmission overhead. A similar approach is introduced in [99] where a distributed group key agreement scheme is proposed based on the Elliptic Curve Diffie-Hellman (ECDH) protocol [100] and which is adaptable to membership changes in a group.

In [101], Li Gong and Wheeler David J. have proposed a grid-based key distribution scheme to establish shared keys. In their work, the deployment area is divided into a 2-D grid where each node occupies a unique point (i, j) in the 2-D space. Keys and corresponding key identifiers are then generated and pre-loaded onto each point in the 2-D grid. Each node in the network is then given a set of keys that

can be described by a set of lines that passes through the node's position (i, j). To find out the common key with another node, each node needs to solve a system of linear equation groups to find out the common intersection points they share and the key at the common points will be used to secure the communications. MA. Hamid et al. have further extended this technique to a 3D space in an enterprise mesh network [37, 38] where each node solves a system of linear plane equations to establish a common key. Since keys are also preloaded onto each node, this scheme is not resilient against node capture attack. For better security, it is advisable to change the keys periodically which is a challenging task given the large dynamic nature of the network.

In [102], B. Zan et al. have introduced an unconventional way of establishing secret keys for V2V and V2I communications. It uses symmetric cryptography to secure communications in VANETs. In V2V, two legitimate users establish a secret key by extracting a sequence of common bits from the wireless channel by monitoring the amount of increase or decrease in the Received Signal Strength (RSS) values. This is possible because of channel reciprocity. An eavesdropper is unable to extract the common bits due to the spatial de-correlation of the channel. In V2I, RSUs broadcast seeds through random channel hopping scheme. A vehicle selects some of the seeds collected from RSUs and applies XOR operation to form a secret key. For an eavesdropper to know the secret key, he or she has to know the correct seeds that are used or seeds that are sent out on the same channel and at the same time.

In all the above mentioned schemes, the exchange of keying information is necessary for deriving the common keys which mean that an adversary could send a lot of bogus messages for verifications which make them vulnerable to DoS attacks. Take the DH key exchange proposed in [96] for example, a dishonest node can send huge amounts of public keys which can simply be random numbers so that the other party is compelled to carry out many modular exponentiations in order to compute the shared keys [103]. Similarly, the polynomial-based approach, the matrix-based key establishment approach, the grid-based key distribution scheme and the signal extraction method of establishing shared keys are vulnerable to DoS attacks. Additionally, the schemes in [86, 87, 88, 37, 38] are susceptible to the node capture attacks because the adversary can compromise a node physically to extract the preloaded

keys.

2.4.2 Asymmetric Cryptography

In asymmetric cryptography, two keys are defined where the private key is used for signing and decrypting a message while the public key is used for signing and encrypting the message. All the cryptography primitives depend on these keys to provide confidentiality, integrity, authenticity and non-repudiation properties. To facilitate the management of these keys, a PKI has been proposed in the IEEE 1609.2 [29] and the ETSI TS 102 940 standards [33]. However, deploying a PKI incurs a high latency mainly due to the expensive certificate management process as described in section 2.2.2.2. To resolve this issue, several approaches have been proposed which can be generalized into two types, namely certificate-less PKI and reducing the CRL checking process. In certificate-less PKI, the basic idea is to eliminate the use of digital certificates entirely so that expensive operations related to certificate management are avoided. In the following, we are going to look at security solutions from these two areas that improve upon the PKI.

2.4.2.1 Certificate-less PKI

Before the birth of PKI, the concept of maintaining a public file to store encryption keys is briefly discussed by Diffie-Hellman when they introduced the Public Key Cryptography (PKC) system. As stated in [97], the encryption key which is the public key used in PKC is made public by storing them in a public file directory along with the users' attributes. Any user can then access the public file directory to retrieve the public key of the user he or she wants to communicate with and encrypt the message. Only the intended receiver with the correct private key can decrypt to reveal the message. This idea was proposed for VANETs by [104] and it is called the Public Key Regime (PKR) scheme. The purpose of PKR is to reduce the cost of managing certificates.

In this model, each RSU in the network is given a public key directory by the Trusted Authority (TA). This public key directory contains the vehicles' IDs and their corresponding public keys certified by the TA. To start a communication in the network, a vehicle has to broadcast its identity to the other communicating vehicle and then the receiving vehicle sends a public key query message to a nearby

RSU to retrieve the sender's public key. Evaluation results from this paper show that the security overhead of the PKR scheme is lower than in certificate-based PKI because the certificate is eliminated from the transmission. It also has a lower latency compared to certificate-based PKI because time-consuming certificate processes are eliminated. In addition, there is no need to download huge CRL making it very scalable in a large scale environment. However, the PKR scheme is designed without any security considerations. Since the RSU needs to service many query requests from the vehicles in a region, this may result in a bottleneck at the RSU leading to a DoS attack which can easily compromise on the availability of service. Moreover, in the PKR scheme, all the RSUs maintaining the public key directory are given the TA's signing key which exposes some security risks.

In [105], Mahmoud Al-Qutayri et al. have proposed a non-PKI approach that relies on the concept of ID-Based Cryptography (IBC) [106] to provide security and privacy in VANETs. In [105], IBC is realized based on the properties of bilinear pairings on elliptic curves formulated by Boneh and Franklin [107]. In [108], D.He et al. have proposed an ID-based scheme that does not rely on bilinear pairings to provide mutual authentication and privacy protection in VANETs. Their method is based on the Schnorr's signature scheme [109]. The idea of IBC is that the public key is the user's identity which could be the email address, network address or other attributes that uniquely identify the user. Consequently, it is not necessary to store, issue and verify the certificates which outperform the certificate-based PKI in terms of latency and communication bandwidth. However, the concern of IBC is the inherent key escrow problem i.e. the TA has possession of the user'private keys. If the TA is compromised, then it would be able to decrypt and issue signatures on behalf of other users through impersonation. Moreover, in the event that the private key gets compromised, the corresponding public key which is bound to a user's identity must be invalidated and re-assigned. Re-assignment of the public key may not impractical in a public network such as V-Mesh networks as it may lead to the user having multiple identities and thus, violating the non-repudiation property in the event of liability.

2.4.2.2 Reduced CRL Checking

AH. Salem et al. have proposed a dynamic key distribution protocol for a PKI-based VANET that leverages on the RSUs to issue and revoke certificates [110, 111]. Each time a vehicle needs a certificate from the CA, it sends a request to the RSUs, which will then help to forward the request message to the CA. The request message is encrypted using a unique pre-shared key consisting of the Electronic License Plate (ELP) and the Electronic Chassis Number (ECN) that is only known by the vehicle and the CA. Thus, the intermediate RSUs that help to forward the message will not be able to read it. Similarly, the certificate issued by the CA back to the requesting vehicles can only be revealed by the requesting vehicle itself. By obtaining the certificates on demand, there is no need to preload a large number of long term and short term certificates in a vehicle which saves storage space. In addition to storage space improvements, their scheme also benefits from a more efficient revocation process through the deployment of RSU managers into the PKI hierarchy. The RSU managers maintains the status of the RSU that assist in the request for the vehicle. If a certificate needs to be revoked, the CA will send a revocation message to the RSU manager which then directs the revocation message to the corresponding RSU where the revoked vehicles resides. The RSU will then send the revocation message to revoke the affected vehicle and also to all the vehicles in its region to inform them. In doing so, the revocation process is greatly improved as compared to broadcasting the entire CRL throughout the network or dividing it into compressed parts and broadcast them. However, since the vehicle has to send a certificate or key request to the CA via the RSUs and RSU managers, the latency to obtain the certificate may be high if the network is very dense.

Papadimitratos et al. have proposed several revocation protocols for VANET to improve the CRL distribution process [26]. The first method is to send a message signed using the CA's private key to the HSM of the vehicle. These messages are used to delete all valid certificates from the vehicle, and preventing the HSM from signing messages in future. This method assumes that there is an infrastructure of RSUs to send the messages to the HSM. It also assumes that the HSM will remove the certificate as instructed. However, this is not necessarily a valid assumption, since the HSM may have been compromised. Furthermore, to ensure that messages from a vehicle is not considered valid, CRL must still be distributed. The second

method is to reduce the size of CRL by a lossy compression scheme(e.g. using a Bloom filter). However, compression increases the number of false positives because the Bloom filter is a probabilistic function. Moreover, this method requires the vehicles to communicate with the CA through the RSUs. The third method is to encode the CRL into multiple self-verifiable pieces using erasure encoding. The pieces are then sent to the RSU for broadcast. When the vehicle receives a piece, it validates and stores it. After sufficient pieces are gathered, the vehicle decodes the CRL.

In [112], an Expedite Message Authentication Protocol (EMAP) has been proposed to overcome the problem of the long delay incurred in checking the revocation status of a certificate using a CRL. EMAP is based on the HMAC where the key used in calculating the HMAC for each message is shared only between unrevoked vehicles. The idea is to verify the HMAC first. If the HMAC is correct, the vehicle proceeds to verify the digital certificate against the CRL. On the other hand, if the HMAC is incorrect, it means that the vehicle who sends the message has been revoked. Therefore, the packet will be dropped and no verification of the digital certificate is performed. When there is a revocation, the HMAC key must also be updated among the nonrevoked vehicles. This is accomplished dynamically using the probabilistic key sharing mechanism. In doing so, the HMAC that is attached to any message subsequently is always guaranteed to come from an unrevoked vehicles. The problem with this scheme is that bilinear pairing operations are expensive. Furthermore, the setting up of the new HMAC key involves a lot of communication overhead if the vehicle misses a number of previous revocation messages.

Table 2.4: Summary of key management solutions

Scheme	**Type**	**Mechanism**	**A**	**B**	**C**
Probabilistic key [86, 87, 88]	Key distribution	Preload key	✗	✓	✗
Polynomial-based approach [89, 90, 91]	Key distribution	Dynamic key generation	✗	✗	✓
Matrix-based key establishment [92, 93, 94, 95]	Key distribution	Dynamic key generation	-	✗	✓

A: Certificate-less, B: Resilient against DoS Attack, C: Resilient against Node Capture Attack

Continued on next page

Table 2.4 – *Continued from previous page*

Scheme	Type	Mechanism	A	B	C
DH key exchange [96, 99]	Key distribution	Dynamic key generation	✗	✗	✓
Grid-based key distribution [101, 37, 38]	Key distribution	Preload	-	✗	✗
Signal extraction [102]	Key distribution	Dynamic key generation	✓	✗	✓
PKR scheme [104]	Certificate-less PKI	Maintain a repository at each RSU	✓	✗	-
Mahmoud Al-Qutayri et al. [105]	Certificate-less PKI	ID-based using IBC	✓	✗	-
D. He et al. [108]	Certificate-less PKI	ID-based using IBC without bilinear pairings	✓	✗	-
AH. Salem et al. [110, 111]	Reduce CRL check	Rely on RSU to distribute revocation	✓	✗	-
Papadimitratos et al. [26]	Reduce CRL check	Revocation of HSM, Compressed CRL	✗	✗	-
EMAP [112]	Reduce CRL check	HMAC using key shared among unrevoked vehicles	✗	✗	-

A: Certificate-less, B: Resilient against DoS Attack, C: Resilient against Node Capture Attack

2.5 Conclusion

In this chapter, we explore two aspects of security solutions for V-Mesh networks namely, the trust model for routing and the key management. We conclude that both trust models and key management systems are essential ingredients in providing a complete security against internal and external attackers. More specifically, trust models are used to counteract selfish misbehaviors launched by internal attacks. On the other hand, cryptographic-based solutions are deployed to defend against unauthorized external attackers and secure the communications against the confidentiality, integrity, authenticity and non-repudiation attacks.

In terms of the trust models, we highlight the advantage and disadvantage of gathering recommendations from the other nodes. We further discuss the various

techniques of aggregating direct and indirect trusts together including the methods of handling the recommendations in these existing schemes. Based on our review, most of the trust aggregation techniques are biased because these approaches favor recommendations that are closely related to their own observations. Other approaches consider only good recommendations from reputable recommenders which make these schemes restrictive. These limitations that inspire us to research on trust aggregation techniques for providing a fair and just trust evaluations.

In our review of the trust models, we also highlight that overhearing is a prevalent technique employed in most trust models by providing a vast review of literature covering works in VANETs, MANETs and WMNs. We also discuss the limitations of overhearing that could affect the accuracy of the results. We conclude that most of the existing schemes have not considered the drawbacks of overhearing in their proposed solutions. Therefore, it is imperative that future systems must have countermeasures to improve it.

To address the limitations of PKI, we classify the solutions into three broad categories namely, key distribution, certificate-less PKI and simplifying the CRL checking process. First, the key distribution approach solves the complexities of PKI by relying on symmetric cryptography to encrypt messages in a communication. This reduces the dependence on asymmetric cryptography, which is considered to be more expensive than symmetric cryptography. Second, the certificate-less PKI avoid the dissemination of certificates during communication so that expensive CRL management is eliminated. Third, to reduce the CRL checking process, additional steps are introduced to first verify a hash value before expensive operations are spent on verifying the certificates and digital signatures. Lastly, we discuss the drawbacks of each scheme to complete our review.

Chapter 3

An Unbiased Trust Model

In this chapter, we propose an unbiased trust model to defend against badmouthing and ballot-stuffing attacks. The novelty of our design lies in the use of DST [113] to characterize the trust modeling and trust aggregation. By using DST, ambiguous recommendations are scaled down appropriately and conflicting trust opinions are resolved, thereby mitigating badmouthing and ballot-stuffing attacks. Furthermore, our approach benefits from not having to implement elaborate and challenging threshold-based filtering design to filter out recommendations which may sometimes lead to a poor judgment. As our model is developed based on the DST, our model is called the DS-Trust model.

The rest of this chapter is organized as follows. Section 3.1 introduces the DST concept. Section 3.2 describes the DS-Trust model in details. Section 3.3 presents the performance improvements of DS-Trust against the badmouthing and ballot-stuffing attacks. Section 3.4 presents the simulation results to demonstrate the performance of DS-Trust. Finally, section 3.5 concludes this chapter.

3.1 Preliminaries

In this section, we identify our design goals, highlight the motivations of using DST [113] and discuss the key concepts which serve as the basis of our proposed DS-Trust model.

3.1.1 Design Goals

The primary objective of DS-Trust model is to solve the routing problem in V-Mesh networks. As mentioned previously, communications between the vehicles and the roadside installations take place over several hops in the network. Therefore, the reliability of communication depends mainly on the routing protocol and whether the nodes are cooperative in forwarding. To this end, the aim of DS-Trust is to help the underlying routing protocol determine a secure end-to-end path free of selfish nodes to provide efficient and reliable communications. In addition, the DS-Trust model is designed to detect misbehaving nodes that attempt to demote or promote the reputation of other nodes in the network by providing false trust ratings. To summarize, the following attacks are addressed by DS-Trust.

- **Blackhole attacks**
  Blackhole is a packet dropping attack where the malicious node behaves selfishly by dropping 100% of the data packets it receives.

- **Grayhole attacks**
  Grayhole is a variant of the blackhole attack where the packet dropping rate is not 100% but occurs with some selective probability.

- **Badmouthing attacks**
  Badmouthing attack refers to the case where a malicious node deliberately provides bad trust recommendations to frame up good peers so as to decrease their trust ratings.

- **Ballot-stuffing attacks**
  Ballot-stuffing attack refers to the case where a malicious node deliberately provides good trust recommendations for malicious peers so as to keep their trust ratings above certain threshold to avoid detection and to improve their chances of being selected as forwarders.

For information on these attacks, we refer the readers to section 2.2.1 where a formal definition is provided. We note here that we do not consider other general attacks such as privacy attacks, impersonation and sybil attacks but refer to the techniques discussed in [39, 5, 114, 115, 116] to defend against them.

3.1.2 DST Background

DST is a theory of evidence based on belief functions and plausible reasoning. It was first developed by Dempster [117] and later formalized into a mathematical framework by Shafer [113] for modeling uncertainty and merging multiple evidences. Unlike traditional probability theory, DST allows for explicit representation of uncertainty.

To illustrate this idea, let us assume that node X is trustworthy as a forwarding node with a belief of 0.8. If traditional probability theory is employed, the remaining probability of 0.2 must be assigned to the untrusted state to satisfy the additive rule of probability. In contrast, DST classifies the remaining belief as uncertainty which means that a node could either be viewed as trustworthy or untrustworthy. Such argument is reasonable since there is no evidence to justify that node X is untrustworthy. By introducing uncertainty into the system, false detection caused by premature accusation can be prevented. In addition, DST is able to handle conflicting evidences that are caused by imperfect observations, misbehaviors or varied users' experiences in the network. To address such conflicts and inconsistencies, DST has proposed a useful operator called the Dempster's rule of combination where agreement among multiple evidences is enhanced through a normalization factor.

For these reasons, DST is highly applicable for modeling trust relationships and trust aggregation, especially in situations where trust opinions from the other peers in the network are contradicting due to misbehaviors. Hence, DST is regarded as a better alternative to traditional probability theory as far as clarity and expressiveness of opinions are concerned. In the following, we discuss three functions related to DST.

3.1.2.1 Basic Probability Assignment

Let Θ be the frame of discernment containing all the possible hypotheses of an environment whose belief is to be assessed. The hypotheses in Θ are mutually exclusive and exhaustive. Let 2^{Θ} be the power set which is an extension of the frame of discernment containing all the subsets of Θ and the empty set ϕ. By defining a power set, an evidence source can distribute a basic probability assignment

(bpa) to not only singleton hypotheses but also to any combinations of singleton, thereby increasing the flexibility of measures including expanding the scope of the opinion space. Suppose j is a finite set of independent evidence sources defined by $j \in \{1, \ldots, J\}$. The bpas generated by each evidence source can be represented by a mass function $m_j(s)$ which is a mapping of a subset $s \in 2^{\Theta}$ to a non-negative value between 0 and 1 given by (3.1).

$$m_j(s) : 2^{\Theta} \rightarrow [0, 1], \quad \forall s \in 2^{\Theta} \tag{3.1}$$

$$m_j(\phi) = 0 \tag{3.2}$$

$$\sum_{s \in 2^{\Theta}} m_j(s) = 1 \tag{3.3}$$

The bpa denotes the amount of belief committed to a particular subset of 2^{Θ} under consideration. Two conditions must be adhered to when allocating bpas, i.e. no bpa should be assigned to the empty set ϕ and the sum of belief mass should be equal to 1 according to (3.2) and (3.3) respectively.

3.1.2.2 Belief Function and Plausiblity Function

The DST framework further defines two functions that make use of the bpas. They are called the belief function and the plausibility function. More formally, the belief function is defined as the sum of all the bpa that directly supports a hypothesis. It represents the degree of confidence that the evidence provided by an evidence source j supports the proposition in question.

$$belief_j(s) = \sum_{p \subseteq s} m_j(p) \quad \forall p \in 2^{\Theta} \tag{3.4}$$

On the other hand, the plausibility function defined in (3.5), encompasses all the bpas that partially or fully support a hypothesis, that is, all the bpas whose interaction with the hypothesis s is not empty are considered as potential beliefs. It can be interpreted as optimistic belief in s. Together, the belief and plausibility function define a belief interval $[belief(s), plausibility(s)]$ where belief function forms the lower boundary and the plausibility function forms the upper boundary respectively.

$$plausibility_j(s) = \sum_{p \cap s \neq 0} m_j(p) \quad \forall p \in 2^{\Theta} \tag{3.5}$$

3.1.2.3 Dempster's Rule of Combination

Given a frame of discernment, it is possible for multiple sources to provide their evidence. Assuming that the evidence comes from independent sources, they can be combined in a pairwise manner using the Dempster's rule of combination defined in (3.6) .

$$m_{DS}(s) = m_1(s) \oplus m_2(s) \oplus \cdots \oplus m_J(s) \quad \forall s \in 2^{\Theta} \tag{3.6}$$

where $m_{DS}(s)$ denotes the resulting mass function after combination and $\oplus$ represents the combination operator. If there are only two evidence sources, E_i and E_j, the pairwise combination operator can be written as:

$$m_{i,j}(s) = \frac{1}{1-K} \sum_{p \cap q = s} m_i(p) m_j(q) \quad s \neq \phi \tag{3.7}$$

$$K = \sum_{p \cap q = \phi} m_i(p) m_j(q) \tag{3.8}$$

where $m_i(p)$ and $m_j(q)$ are bpas provided by evidence sources, E_i and E_j respectively. The quantity K denotes the amount of conflict between the two evidence sources whose interaction is an empty set. The denominator $(1-K)$ serves as the normalization factor to ensure that the total sum of combined masses, $m_{i,j} = 1$. In essence, the DST's rule of combination sums up all the evidences that support each other and suppresses the conflict in the system through the normalization factor.

3.2 DS-Trust Model

In this section, we describe the five modules of the DS-Trust model shown in Figure 3.1 and their relationship to each other. These five modules are installed on every node which can be a vehicle or RSU in the architecture and they consist of the monitor module, the feedback module, the correlation module, the fusion module and the decision module.

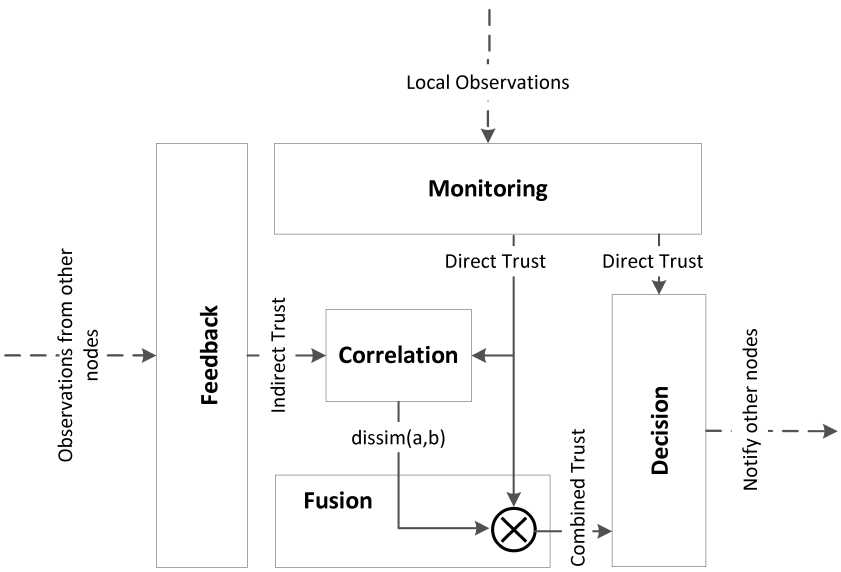

Figure 3.1: DS-Trust model.

3.2.1 Monitoring Module

The monitor module is equipped with the overhearing mechanism [36] to monitor the forwarding behavior of the next hop. It helps to keep track of the number of packets sent and the number of packets overheard locally. More specifically, each node stores the packet ID and the node ID that the packet is directed to in a table. When the node overhears a packet sent and finds a match in the corresponding table, the table entry corresponding to the overheard packet ID is deleted. During each trust monitoring period denoted as T, the node tallies the sum in the table and computes the forwarding probability of the downstream node as in (3.9). Next, the forwarding probability of a node is converted into a trust value using a set of equations (3.10) and (3.11).

$$p_f = \frac{\#\, of\, overheard\, packets\, sent\, by\, n_i}{\#\, of\, packets\, sent\, to\, n_i\, for\, forwarding} \tag{3.9}$$

$$DT = \begin{cases} 1 - 0.5H_b(p_f), & \text{for } 0.5 \leq p_f \leq 1 \\ 0.5H_b(p_f), & \text{for } 0 \leq p_f < 0.5 \end{cases} \tag{3.10}$$

$$H_b(p_f) = -p_f \log_2(p_f) - (1 - p_f)\log_2(1 - p_f) \tag{3.11}$$

where p_f denotes the forwarding probability of a node, DT denotes the trust value of a node and $H_b(p_f)$ denotes the binary entropy function. The use of the entropy

function in (3.11) is to reflect the amount of uncertainty about p_f since the promiscuous mode of monitoring is affected by channel conditions and transmission power [36]. The mapping function in (3.10), on the other hand, is to bind the trust values in the interval $(0, 1)$ where a trust value of 1 means a node is fully trusted while a trust value of 0 means a node is not trusted. Furthermore, we incorporate the exponential averaging function as shown in (3.12) to give more weight to the recent trust value than the past trust values.

$$DT_t = \alpha \cdot DT_t + (1 - \alpha) \cdot DT_{t-1} \tag{3.12}$$

where α is a constant smoothing factor between 0 and 1, DT_t represents the trust value at time t and DT_{t-1} represents the previous trust value recorded by the monitoring module. If the smoothing factor α is large, it discounts the previous trust faster. Once the direct trust is obtained, it is passed to the correlation module and the decision module for further processing.

3.2.2 Feedback Module

This module is responsible for gathering recommendation trusts from the neighboring nodes. At the end of every trust monitoring period T, the node broadcasts a request message for the direct trust of a target node. When the neighboring nodes receive the request message, each of them checks its own record to see if there is a trust record for the target node. If a record is found, the neighboring nodes send a recommendation message containing the trust value of the target node back to the requestor. In this case, the recommendation trusts originating from the recommenders are calculated the same way as described in the monitoring module. When the requestor receives the recommendation message, it weighs the received recommendation trust value by its opinion of the recommender to calculate the indirect trust and then, outputs it to the correlation module. Suppose node C is the target node and node A is the requestor soliciting the opinion of the recommender B, the indirect trust denoted by IDT is formulated as (3.13).

$$IDT_{AC} = DT_{AB} \cdot DT_{BC} \tag{3.13}$$

3.2.3 Correlation Module

The correlation module and the fusion module *(to be discussed next)* are the two cornerstones of the DS-Trust model. They are designed to help mitigate bad-mouthing and ballot-stuffing attacks. Unlike existing trust models [51, 53, 54, 55, 56, 58] presented in section 2.3.1.2, every single recommendation trusts received by the correlation module are considered in the trust aggregation irrespective of whether they are good or bad. To process these recommendations, the correlation module employs the dissimilarity test to compare the direct trust record and the received indirect trust record. The dissimilarity ratio measures the amount of conflict between one's own opinion and the opinions of others. It determines how much the indirect trust records contribute towards the final trust aggregation. The dissimilarity ratio, as shown in (3.14), is expressed as the normalization of the absolute difference between the two trust records. If the dissimilarity ratio is large, it implies that the two trust records are in conflict. Thus, the evidence supporting the indirect trust record is viewed as uncertainty. On the other hand, a small deviation means that the two trust records are almost similar and this should amplify the belief given by reduced uncertainty about the observed proposition.

$$dissim(a, b) = \frac{|a - b|}{|a| + |b|} \tag{3.14}$$

3.2.4 Fusion Module

The role of the fusion module is to evaluate the trust value of a node by fusing the direct trust and indirect trust together. Before the actual aggregation takes place, the results of the dissimilarity test are used to re-evaluate the contribution of the indirect trust values and this is carried out based on the DST [113]. We first classify the behavior of the nodes into two states: trusted (T) and untrusted ($\bar{T}$) which forms the frame of discernment Θ defined as $(T, \bar{T})$. For the power set denoted by 2^{Θ}, it consists of the following subsets:

$$2^{\Theta} = [(T), (\bar{T}), (T, \bar{T}), \phi] \tag{3.15}$$

The set represented by $(T, \bar{T})$ denotes uncertainty in our model which means that a node can be trusted or untrusted. For classification of direct trust, we apply

the direct trust values obtained from (3.10) or (3.12) as the bpas to denote the strength of evidence associated with a particular subset of 2^{Θ} and define the following classification rules in (3.16) and (3.17) to assign mass functions.
If DT $\geq \gamma$, then

$$\begin{aligned} m(T, \bar{T}) &= 1 - DT \\ m(T) &= DT \\ m(\bar{T}) &= 0 \end{aligned} \tag{3.16}$$

Else

$$\begin{aligned} m(T, \bar{T}) &= 1 - DT \\ m(\bar{T}) &= DT \\ m(T) &= 0 \end{aligned} \tag{3.17}$$

In the design, if the trust value is above a certain detection threshold called γ, it will be classified as a trusted node, whereas a trust value less than γ will be classified as an untrusted node. As an example, if the threshold γ is 0.5 and the direct trust value of a node is 0.6, the bpa assigned to the set $m(T)$ will be 0.6. The remaining belief mass of 0.4 will be allocated to the set $m(T, \bar{T})$. For classification of indirect trust values, we leverage on the $dissim(a, b)$ value received from the correlation module and classify the node according to rules defined in (3.18) and (3.19).
If IDT $\geq \gamma$, then

$$\begin{aligned} m(T, \bar{T}) &= dissim(a, b) \\ m(T) &= 1 - m(T, \bar{T}) \\ m(\bar{T}) &= 0 \end{aligned} \tag{3.18}$$

Else

$$\begin{aligned} m(T, \bar{T}) &= dissim(a, b) \\ m(\bar{T}) &= 1 - m(T, \bar{T}) \\ m(T) &= 0 \end{aligned} \tag{3.19}$$

From (3.18) and (3.19), the dissimilarity value is treated as uncertainty for the set $m(T, \bar{T})$. This is in line with the principles of DST since the amount of conflict between two trust records denotes uncertainty. As the deviation between the direct and

indirect trust increases, the belief mass pertaining to the uncertain event increases accordingly. Next, depending on the value of the received IDT, the bpas for the set $m(T)$ and $m(\bar{T})$ are updated accordingly such that $m(T) + m(\bar{T}) + m(T, \bar{T}) = 1$. Once the bpas have all been updated for the direct and indirect trust, the Dempster's rule of combination is applied to fuse the two trust evidences together and the result is sent to the decision module for actions. In the following, we provide a numerical example to illustrate the trust aggregation process.

Example: Let us suppose that the first evidence E_1 which represents node A's direct trust value of node C is 0.9. Since it is more than the detection threshold γ of 0.5, a bpa value of 0.9 is assigned to the set $m(T)$ and the remaining 0.1 is assigned to the uncertainty set $m(T, \bar{T})$. Suppose now the IDT about node C from one of the recommenders, say node B is 0.1. The dissimilarity ratio between A's trust and B's trust will be 0.8 according to (3.14). Thus, for the second set of evidence, E_2 which represents the indirect trust value of node C based on B's recommendation, $m(\bar{T})$ will be assigned with 0.2 since IDT is less than γ and the remaining 0.8 will be allocated to the set $m(T, \bar{T})$. With this information, we tabulate Table 3.1 and use the Dempster's rule of combination in (3.7) and (3.8) to evaluate the final trust.

Table 3.1: Aggregation of evidence source E_1 and evidence source E_2

E_1 \ E_2	$\{\phi\}$=0	$\{T\}$=0	$\{\bar{T}\}$=0.2	$\{T, \bar{T}\}$=0.8
$\{\phi\}$=0	$\{\phi\}$=0	$\{T\}$=0	$\{\bar{T}\}$=0	$\{T, \bar{T}\}$=0
$\{T\}$=0.9	$\{T\}$=0	$\{T\}$=0	$\{\phi\}$=0.18	$\{T\}$=0.72
$\{\bar{T}\}$=0	$\{\bar{T}\}$=0	$\{\phi\}$=0	$\{\bar{T}\}$=0	$\{\bar{T}\}$=0
$\{T, \bar{T}\}$=0.1	$\{T, \bar{T}\}$=0	$\{T\}$=0	$\{\bar{T}\}$=0.02	$\{T, \bar{T}\}$=0.08

Using the Dempster's rule of combination, the combined belief $m_{1,2}(T)$ that node C is trusted from node A's perspective is

$$
\begin{aligned}
m_{1,2}(T) &= \frac{1}{1-K}[m_1(T) \cdot m_2(T, \bar{T})] \\
&= \frac{1}{1-0.18}(0.72) \\
&= 0.87805
\end{aligned}
$$

3.2.5 Decision Module

If the network is sparse and there are no recommenders to provide indirect trust, the decision module will base its decision only on the direct trust gathered from the monitoring module. Otherwise, the combined trust value as calculated per the fusion module is used. Below summarizes the actions undertaken by the evaluating node when the trust value falls below the detection threshold γ.

- Isolate selfish nodes - the evaluating node will blacklist the misbehaved node. At the same time, it conducts a blacklist broadcast throughout the network to inform other nodes who further block it from all future communications. In our approach, we choose to exclude the blacklist nodes permanently from routing as it is better than penalizing them and allowing them to regain their trust slowly. The latter may induce the misbehaved nodes to misbehave on and off intermittently to avoid detection which is a more challenging task to solve. The node who has been blacklisted but wishes to re-join the network may contact an authorized operator to be reinstated. The operator will then track the number of times a particular node has been blacklisted. If a particular node has been blacklisted repeatedly beyond a certain number of counts pre-defined by the operator, it can never re-join the network. By relying on a trusted authority, the operator has a better visibility of the health of the nodes and ensuring that only good nodes can participate in the network.

- Initiate new route discovery - information about the misbehaved node is also sent to the underlying routing protocol which will trigger the routing protocol to send a Route Error (RERR) message to notify the source node. Subsequently, a new route discovery will be initiated to find a path free of selfish nodes.

3.3 Security Analysis

We conduct an experiment to analyze numerically the changes in the trust value as a function of increasing badmouthing and ballot-stuffing attackers and compare our results to the benchmarking schemes discussed in section 2.3.1.2. The benchmarking schemes used are the linear opinion pooling technique, entropy-based probability model, subjective logic and the regression analysis technique based on the works of [71, 55, 80] and [82] respectively.

The aim of this comparative study is to validate the effectiveness of the existing trust aggregation schemes in comparison to using DST for mitigation of badmouthing and ballot-stuffing attacks. In our comparison with the benchmarking schemes, the main focus is the idea put forth in the related papers. As such, in the implementation of the linear opinion pooling technique presented in [71], we are not concerned about the optimal weight selection and therefore, have configured the weights for both the direct and indirect trust components of the linear combination function to 0.5. Also in [80], the key point is to assess the effectiveness of using the subjective logic operators to combine trust, instead of validating the fusion of node-centric and data-centric opinions from different detection mechanism as presented in the paper. Nevertheless, we try to model as closely as possible in accordance with each paper to ensure a fair comparison.

3.3.1 Experimental Setup

We consider the scenario as shown in Figure 3.2 to illustrate the aggregation of direct trust and indirect trust. We are interested in the number of badmouthing and ballot-stuffing attackers that can swing the trust values into the untrusted region. For this purpose, we set the trust detection threshold as 0.5.

According to Figure 3.2, node A has local observations about node B which is known as the direct trust. Apart from that, we assume there are up to 20 recommenders (node C to V) from which node A can gather recommendation trusts about node B. The direct trusts are denoted by solid lines, whereas the recommendation trusts are represented by dotted lines. When node A receives the recommendation trusts, it calculates the indirect trusts by weighing the recommendation trust values based on the trust level of the recommenders C,D,$\cdots$,V etc. We assume that

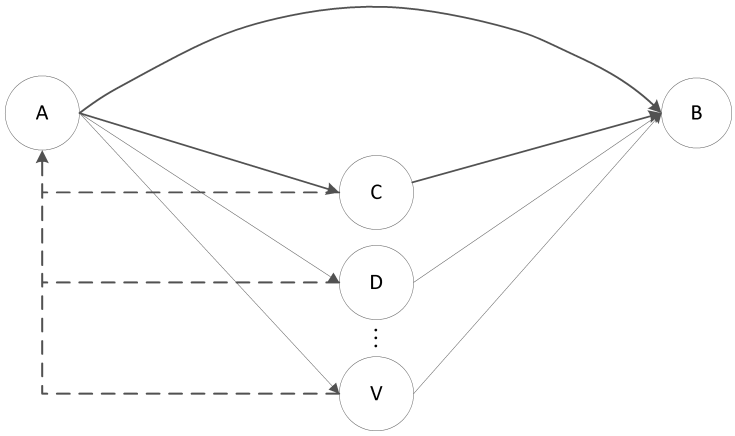

Figure 3.2: Trust aggregation.

node A has the same direct trust value on each of the recommenders so that any observed changes in the trust aggregation results are due to the difference in the recommendation trust values.

In the first experiment, we start by configuring one recommender to send a low rating of 0.1 to node A to simulate badmouthing attacks and increase the number of badmouthing recommenders each round until it reaches 20. The same rule applies to the second experiment except that each recommender is now configured to feedback a rating of 0.9 to simulate ballot-stuffing attacks. In this comparison, the trust value is defined as a continuous value in the range (0,1). As such, the trust value of the entropy-based probability model [55] used for benchmarking is re-mapped into the range (0,1) using equation (3.10) for fair comparisons.

3.3.2 Resiliency to Badmouthing Attacks

Figure 3.3 describes the changes in the aggregate trust value by varying the number of badmouthing attackers. As seen in Figure 3.3, the trust value of the regression analysis approach, the subjective logic model and the linear opinion pooling technique drops drastically down into the untrusted region when there is only one badmouthing attacker in the network. The entropy-based probability model is slightly better as it is able to tolerate up to two badmouthing attackers. In contrast, the aggregate trust value of the DS-Trust model has the best performance. It is able to maintain a high trust within the trusted region for up to ten badmouthing attackers before the trust value enters the untrusted region.

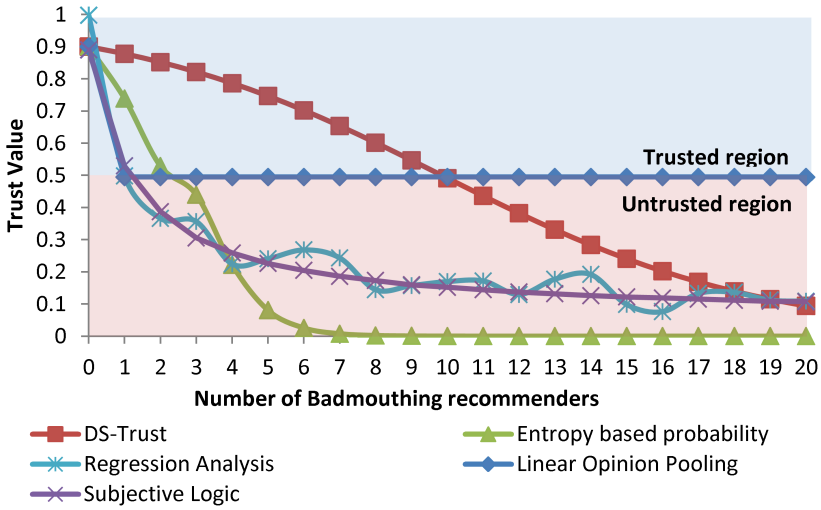

Figure 3.3: Trust value as a function of badmouthing recommenders.

The improvement is due to the treatment of uncertainty when conflicting trust records are received. In particular, the dissimilarity ratio of the two records is treated as uncertainty which is viewed as either trusted or untrusted in DST framework. Subsequently, when the Dempster's rule of combination is applied to combine the direct trust and the indirect trust, uncertainty is absorbed into the aggregation process which amplifies the belief that the node is trusted. Therefore, the DS-Trust model has a slower trust decay compared to the other benchmarking schemes. This shows that the proposed dissimilarity test and the Dempster's rule of combination are able to mitigate the effects of badmouthing attacks effectively.

From Figure 3.3, we further observe that the aggregated trust value based on the linear opinion pooling technique does not span the full range of possible trust values as seen by the low dip just below the trust threshold boundary of 0.5. The aggregated trust value is 0.495 indicating that the node is untrustworthy. This is due to the choice of the weighting factors which we configure as 0.5 in this comparative study. The performance of the linear opinion pooling technique can be improved by placing a heavier weight on the direct trust component of the linear function. However, relying too much on the direct trusts would downplay the influence of the indirect trusts towards the final trust aggregation.

To resolve this problem, R. Chen et al. [71] has proposed to set the weights dynamically in response to past node changes and environmental changes. However, this approach is not indicative enough to model the actual behavior of a node and

can only be regarded as an estimate. The other reason for the small trust variation as seen from the flat curve is that all the indirect trust values are first combined into a single aggregate indirect trust value using the weighted average approach before it is merged with the direct trust value. Therefore, the impact of the aggregated indirect trust is very small in comparison to the direct trust observations.

3.3.3 Resiliency to Ballot-stuffing Attacks

Figure 3.4 shows the trust relationship when there are ballot-stuffing attackers in the network. Similar results are observed where the DS-Trust model is able to tolerate up to ten ballot-stuffing attackers before it succumbs to the false recommendations while the rest of the scheme are vulnerable to a small number of ballot-stuffing attackers. In Figure 3.4, the aggregated trust value based on the linear opinion technique is 0.5 which implies that node B is trustworthy. This value is calculated based on the assumptions that the trustworthiness of each recommender from node A's perspective is 1. We reiterate that the linear opinion approach is not effective even through the weights of the linear combination function can be dynamically adjusted according to the operation profile of a node [71]. Furthermore, a node with an operation profile exhibiting healthy energy level and high cooperative index does not necessarily imply that it is cooperative in nature and would adhere to the rules of the protocol.

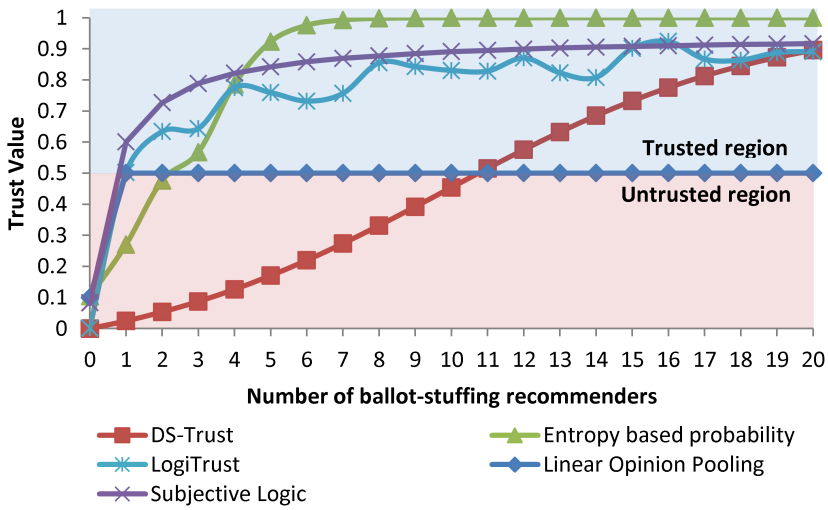

Figure 3.4: Trust value as a function of ballot-stuffing recommenders.

3.4 Performance Analysis

In this section, we incorporate the DS-Trust model into the Ad hoc On-Demand Distance Vector (AODV) protocol [118] and evaluate the performance in terms of the packet delivery ratio, normalized routing overhead and the network throughput. The simulations are performed using the Network Simulator NS-3 (v3.20) [119] and the results are compared to the baseline AODV, and a variant of the DS-Trust without recommendation trusts which is called the DS-Trust (w/o recommendations) model. Lastly, we analyze the computational complexity to evaluate the execution time of our model.

3.4.1 Simulation Environment

We consider two simulation topologies for the performance evaluation of DS-Trust. The first topology is a static environment to mimic the routing in the mesh backbone of a V-Mesh network. In this case, the routing of packets is strictly confined within the RSUs themselves. An example of such usage is I2I communication, where information is passed among RSUs in the WMN backbone to the central ITS subsystem. Other scenarios include propagation of messages between RSUs such as the controlling of traffic signals to give way to emergency vehicles. The second topology is based on a hybrid environment that includes static and mobile nodes to model communications in a V-Mesh network. The mobiles nodes refer to pedestrian, cyclists or vehicle that attempt to establish V2I or P2I communications with the static RSUs in the mesh backbone, and vice-versa. Such communication scenarios are typically used in safety-related applications and value-added applications, such as in collision avoidance applications and Internet access provisioning services. For ease of description, we refer to the first simulation topology as the infrastructure-based V-Mesh network and the latter as the hybrid-based V-Mesh network.

Figure 3.5 shows the infrastructure-based V-Mesh network where the locations of the source and destination nodes are defined at the far left and far right of the square grid and are represented by a darker color. In order to simulate I2I communications, we define 10 Constant Bit Rate (CBR) flows from the source and to the destinations. In the hybrid-based V-Mesh environment depicted in Figure 3.6, 50 mobile nodes are added to the static topology of 100 nodes to simulate movements of pedestrians

or cyclists and we simulate 8 CBR flows from the four gateway nodes located at four corners of the static grid to any random mobile nodes and vice-versa. The starting time of each flow is uniformly distributed between 30 seconds and 200 seconds. The following performance metrics are used to evaluate the proposed DS-Trust model and the rest of the simulation parameters are given in Table 5.4.

- Packet Delivery Ratio (PDR) refers to the ratio of the number of delivered packets to the number of packets generated by the CBR sources.
- Normalized Routing Overhead (NRO) refers to the number of routing control packets, such as the Route Request (RREQ), RREP, RERR and the trust related control packets transmitted per data packet delivered at the destination.
- Throughput refers to the amount of data successfully delivered to the intended destinations over a wireless channel. It is measured in bits per second (bps).
- False positive rate is the ratio of the number of nodes that the DS-Trust model misreports as misbehaving to the total number of nodes in the network.

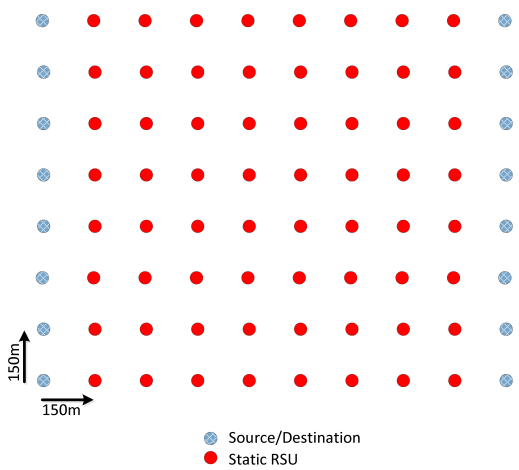

Figure 3.5: Simulation topology for infrastructure-based V-Mesh.

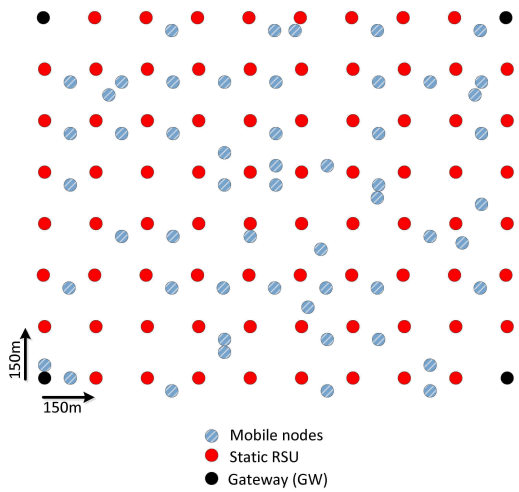

Figure 3.6: Simulation topology for hybrid-based V-Mesh.

Table 3.2: Simulation parameters

Simulation tool	NS-3
Grid spacing	150m
Transmission range	250 m
Network area	1350 m x 1350 m
Data rate	16kbps
Packet size	512 Bytes
Packet generation rate	4 packets/s
Simulation time	300 s
Traffic type	CBR
Transport protocol	UDP
Mac protocol	IEEE 802.11b
Propagation loss model	RangePropagationLossModel
Physical layer	YansWifiPhy channel
Detection Threshold, γ	0.5
Trust Monitoring Period, T	20s
Infrastructure-based V-Mesh network	
Mobility pattern	Static
No. of nodes	100
Traffic	10 source-destination pairs
Routing Protocol	AODV (disable HELLO)
Hybrid-based V-Mesh network	
Mobility pattern	50 mobile nodes, RandomWaypoint
No. of nodes	100 static, 50 mobile nodes
Traffic	8 source-destination pairs
Routing Protocol	AODV with HELLO

3.4.2 PDR Performance

Figure 3.7 and Figure 3.8 show the PDR performance of the various schemes in an infrastructure-based V-Mesh network and a hybrid-based V-Mesh network respectively, when the network is under blackhole attacks. From Figure 3.7, we observe that the PDR of all the schemes remains almost the same when there is no blackhole attacker in the network. As the number of blackhole nodes increases, the PDR starts to decline. However, the DS-Trust model is able to achieve about 30% improvement in the PDR over that of the baseline AODV and about 15% improvement in the PDR over the DS-Trust (w/o recommendations) model.

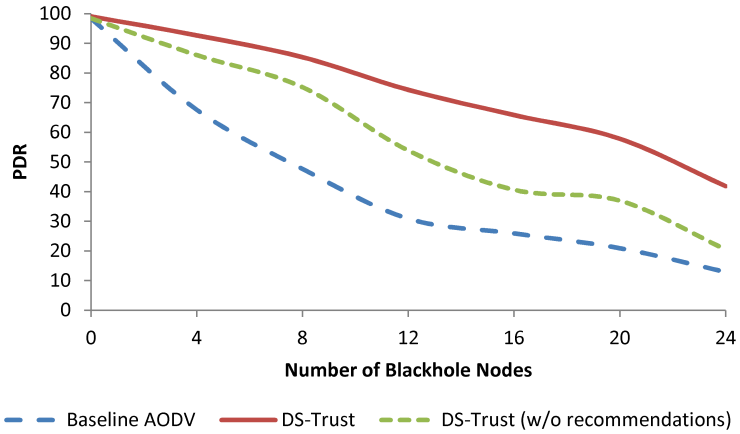

Figure 3.7: PDR in an infrastructure-based V-Mesh under blackhole attacks.

DS-Trust is more superior to the DS-Trust (w/o recommendations) because it gathers recommendation trusts from the neighboring nodes, which improves the detection time of selfish nodes in the network. As the number of blackhole nodes increases further, the PDR of the DS-Trust model starts to decline gradually due to more blackhole nodes being identified and isolated and fewer alternatives are available for the forwarding paths.

Similar results can be observed in the hybrid-based V-Mesh network shown in Figure 3.8 where the PDR improvement for the DS-Trust model is about 14% higher and 10% better compared to the baseline AODV and DS-Trust (w/o recommendations) model respectively. The PDR improvement is lower for the hybrid-based V-Mesh network than in the infrastructure-based V-Mesh network because it is more difficult to maintain link stability due to the node movement. Moreover, the effect

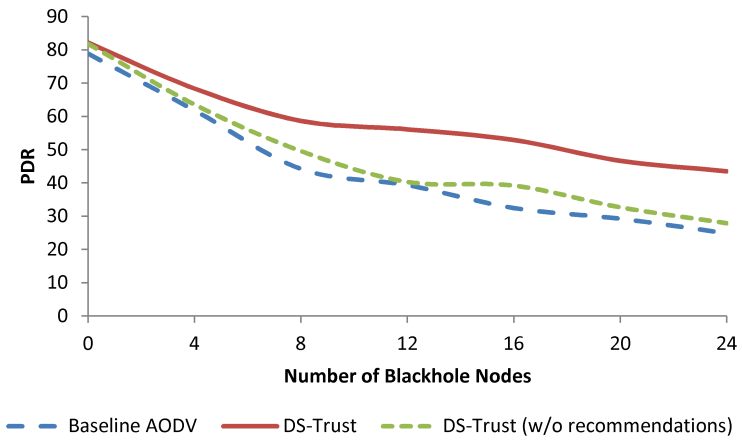

Figure 3.8: PDR in a hybrid-based V-Mesh under blackhole attacks.

of not using the recommendation trusts from the other nodes is more visible in this plot as there is almost no improvement in the PDR when compared to the baseline AODV.

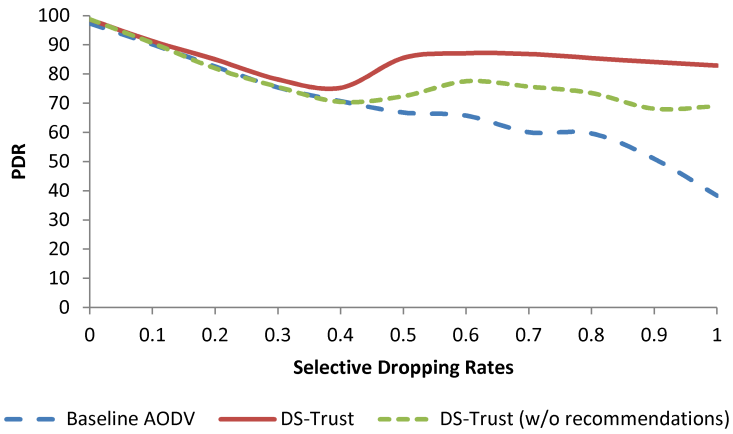

Figure 3.9: PDR in an infrastructure-based V-Mesh under grayhole attacks.

Next, we study the PDR performance of the infrastructure-based V-Mesh network and the hybrid-based V-Mesh network under the influence of grayhole attacks. We assume that 10% of the network nodes are grayhole attackers and they carry out dropping rates between 0% and 100%. Figure 3.9 shows that the PDR performance of the DS-Trust model is almost similar to the baseline AODV when the selective dropping probability is between 0 and 0.4. However, as the dropping rate increases from 0.4 to 0.5, we observe that the PDR of the DS-Trust model begins to increase.

The main explanation for this is that the trust rating in the DS-Trust model is computed using direct and indirect trusts. As more recommendations are available as indirect trusts, the accuracy of the trust evaluation will improve. Hence, DS-Trust model is able to detect and isolate grayhole nodes faster which leads to an improvement in the PDR. In contrast, there is only a slight improvement in the PDR of the DS-Trust (w/o recommendation) because the trust computations are derived based only on direct trusts. As a result, there may still be grayhole nodes that are dropping the data packets in the network. When the dropping rate reaches 0.5, which corresponds to the trust detection threshold, it is clear that the DS-Trust model is superior to the baseline AODV and DS-Trust (w/o recommendations) models. The DS-Trust model is able to improve the PDR to an average of about of 80%. On the other hand, the DS-Trust (w/o recommendations) model can only increase the PDR by an average of 15% because it does not rely on recommendations trusts.

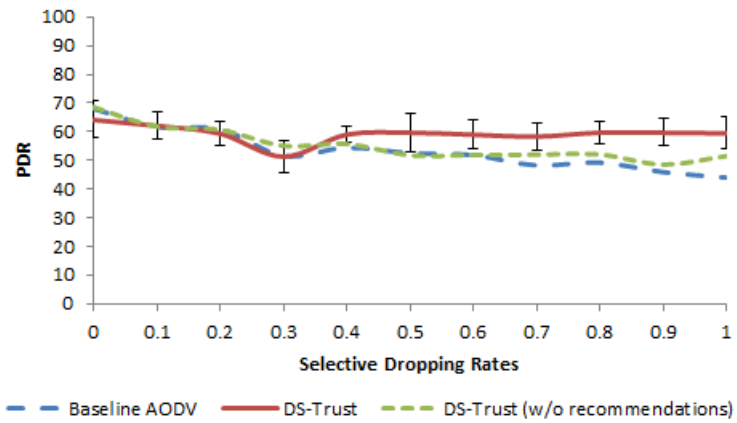

Figure 3.10: PDR in a hybrid-based V-Mesh under grayhole attacks.

Similar results are observed in the hybrid-based V-Mesh network topology. As shown in Figure 3.10, the PDR of the DS-Trust model improves only when the grayhole attackers exhibit 50% or more dropping rate, which corresponds to the trust detection threshold in our model. However, the PDR improvement is only 10% better than the baseline AODV compared to the 28% in the static case. Similarly, the PDR improvement over the DS-Trust (w/o recommendations) model is only about 7%-8% compared to 12%-13% in the static environment. The main reason for the lower PDR improvement is node mobility. When nodes are mobile, the links become

relatively unstable and less reliable which results in more packets drop. In the case of DS-Trust (w/o recommendations) model, the PDR did not improve at all because the recommendation trusts are not disseminated to other nodes to allow them to make a better judgment. We also observe that the PDR of the DS-Trust model decreases at around the dropping rate of 0.3. This is expected because the detection threshold of the DS-Trust model is configured as 0.5. Therefore, the performance of the DS-Trust model should follow the performance of AODV, as if there is no defense mechanism. As the dropping rate of the grayhole nodes starts to increase above 0.3, the PDR of the DS-Trust model also starts to increase. This is because the proposed method of aggregating recommendations with the direct trusts helps to improve the accuracy of the trust computations. Thus, more grayhole nodes will be detected and isolated, which explain the increase in the PDR performance of the DS-Trust model even before the trust detection threshold of 0.5.

3.4.3 NRO Performance

The NRO metric is a measure of the effective utilization of the wireless channel. First, we examine the NRO performance under the blackhole attacks. Next, we analyze the performance in the presence of grayhole attacks. Figure 3.11 compares the NRO performance of the various schemes in an infrastructure-based V-Mesh network.

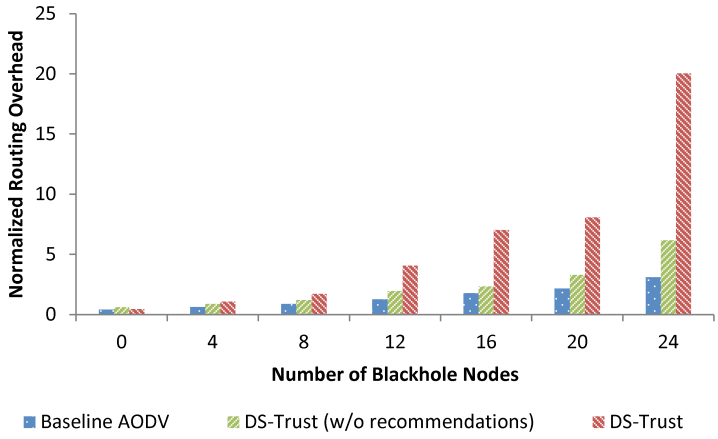

Figure 3.11: NRO in an infrastructure-based V-Mesh under blackhole attacks.

Simulation results show that the baseline AODV has the lowest NRO followed by the DS-Trust (w/o recommendations) model and the DS-Trust model. The NRO performance of the DS-Trust model is higher than the DS-Trust (w/o recommendations) model because of the dissemination of recommendation trusts. In addition, both the DS-Trust models are higher than the baseline AODV because of the extra control packets introduced by our trust model, especially the periodic exchanges of trust information, the broadcast of control messages and the re-initiation of new route discoveries upon detection of blackhole nodes.

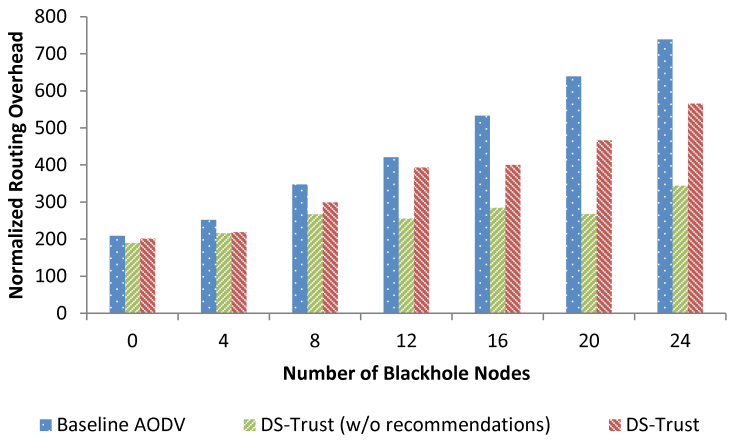

Figure 3.12: NRO in a hybrid-based V-Mesh under blackhole attacks.

On the other hand, in the hybrid-based V-Mesh network scenario as shown in Figure 3.12, the NRO performance of the DS-Trust model is much lower than the baseline AODV compared to the static case in Figure 3.11. This is because when a secure path is found, it is unlikely to change unless the node moves out of transmission range. Even when the link breaks due to mobility, the list of blackhole nodes are circulated to the other neighboring nodes. Therefore, the routing protocol avoids them during a new route discovery which leads to lesser route discoveries and lesser NRO.

In the case of selective dropping or grayhole nodes in an infrastructure-based V-Mesh network, the increase in the NRO is more apparent when the dropping rate is 50% and more. This is true because the trust detection threshold is 0.5. As a result, more control packets are being broadcast to inform the other nodes of the grayhole nodes, including the control packets needed for a new route discovery. On

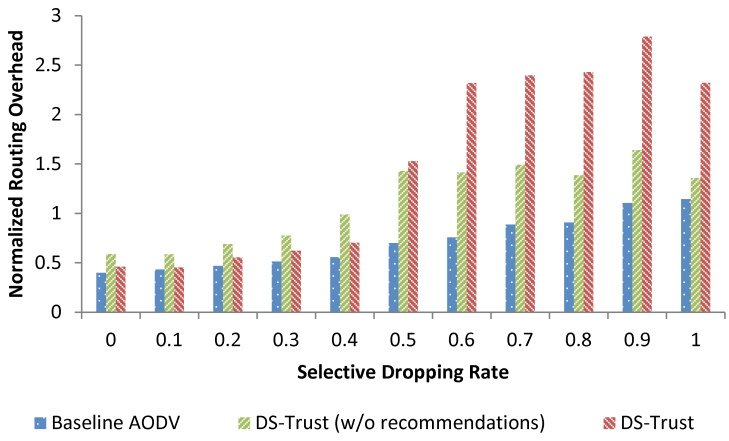

Figure 3.13: NRO in an infrastructure-based V-Mesh under grayhole attacks.

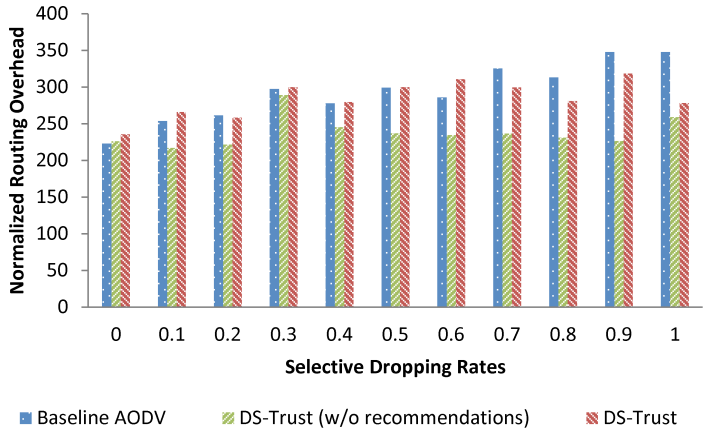

Figure 3.14: NRO in a hybrid-based V-Mesh under grayhole attacks.

the other hand, the NRO performance for both the DS-Trust models in the hybrid-based V-Mesh network are quite similar to the Figure 3.12 where both the NRO performances are lower than the baseline AODV. This is illustrated in Figure 3.14.

3.4.4 Effects of Mobility

Next, we investigate the effects of mobility on the PDR and the NRO performance in a hybrid-based V-Mesh network under the influence of the blackhole nodes. The number of blackhole nodes in the network is 8 and the speed is varied from 10m/s to 40m/s. Simulation results in Figure 3.15 show that the DS-Trust model has the highest PDR performance compared to the baseline AODV and the DS-Trust (w/o recommendations) model. As the node speed increases, the PDR decreases for all the schemes due to more link breakages in the network.

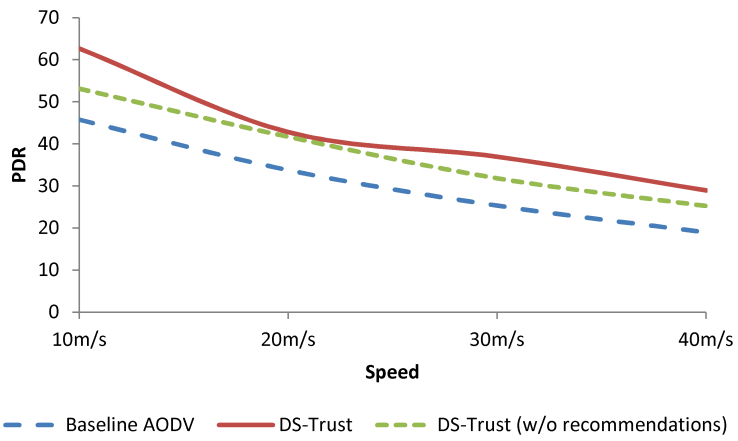

Figure 3.15: PDR in a hybrid-based V-Mesh under varying speed.

Figure 3.16 illustrates the NRO performance for different mobility speeds. It is observed that the NRO performance increases when the node mobility is high. However, the NRO performance of the DS-Trust model is lower than that of the baseline AODV. This confirms our observations that DS-Trust model is able to isolate selfish nodes and exclude them from routing.

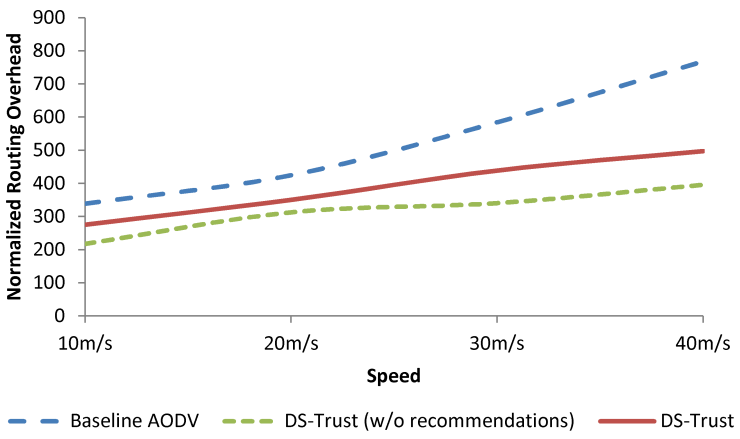

Figure 3.16: NRO in a hybrid-based V-Mesh under varying speed.

3.4.5 Throughput Performance

We are interested in the throughput performance of the DS-Trust model when the application rate is increased. Similar to the previous experiment setting, we consider the performance in an infrastructure-based V-Mesh network and a hybrid-based V-Mesh network. We assume there are 8 blackhole nodes and the application rate is varied from 16384bps to 200000bps.

Figure 3.17 compares the throughput performance of the DS-Trust model in an infrastructure-based V-Mesh network with the baseline AODV and DS-Trust (w/o recommendations) model. As shown in Figure 3.17, the DS-Trust model denoted by the line with the 95% confidence tick marks, is able to achieve a higher throughput than the DS-Trust (w/o recommendations) for application rate between 16384bps and 65536bps. Beyond the rate of 65536bps, there is no performance improvement between the two models.

This could be due to the blacklist of false positives located near the source node that results in no available forwarding paths to send the packets to the destination. We also observe that the throughput for the DS-Trust and the DS-Trust (w/o recommendations) starts to decrease beyond 65536bps. This is caused by the high packet collisions due to the increased sending rate. Since a fixed trust detection threshold is assumed, the DS-Trust models are not able to differentiate losses due to the collisions or malicious intent. Thus, higher false positives are generated, resulting in a reduction in the throughput performance.

The throughput performance of the hybrid-based V-Mesh network is presented

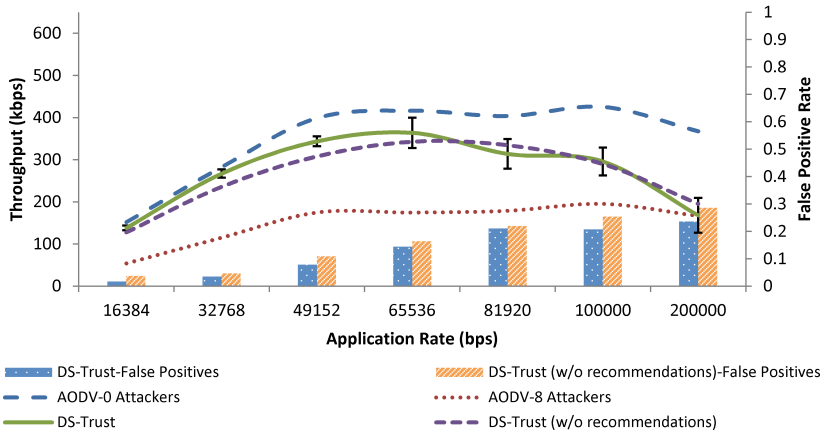

Figure 3.17: Throughput in an infrastructure-based V-Mesh.

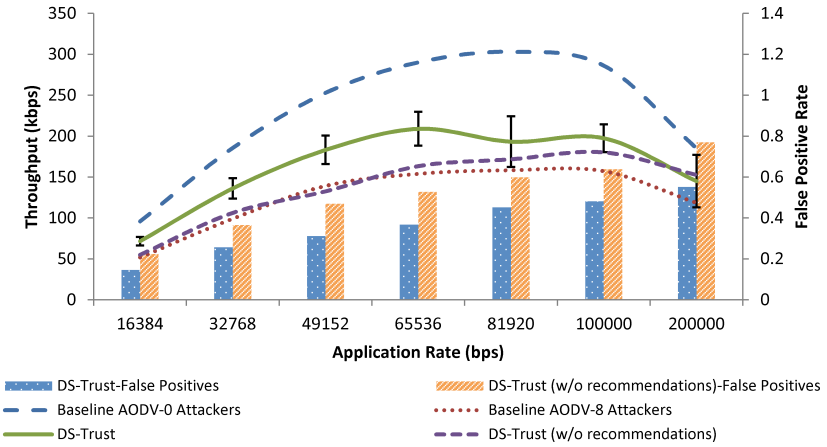

Figure 3.18: Throughput in a hybrid-based V-Mesh.

in Figure 3.18 where it is observed that the DS-Trust model is able to maintain a high throughput throughout the range of the packet sending rates. Comparing the throughput performance of the baseline AODV and the DS-Trust (w/o recommendations) model, the throughput improvement of the DS-Trust is about 25% higher. The throughput of the DS-Trust (w/o recommendations) model remains low because the detection of selfish nodes takes a long time since the recommendation trusts are not used, hence, more packet drop. Furthermore, blackhole nodes may move to other parts of the network to conduct packet dropping attacks. In terms of the false positive rate, DS-Trust model performs better than the DS-Trust (w/o recommendations) model because the trust evaluation is improved using recommendation opinions from many other nodes.

3.4.6 Computational Complexity

First, we estimate the computational complexity of each module in the DS-Trust model using the Big O notation. After that, we merge them to determine the overall execution time of the DS-Trust model.

In the monitoring module, each node needs to compute the direct trust of the downstream nodes it encounters every trust monitoring period. So the computation complexity of the monitoring module is $O(N)$ where N is the number of nodes in the network. The feedback module sends out recommendation requests to solicit recommendation trusts from its one hop neighbors. Upon receiving the recommendation trusts, each node needs to lookup the trust value of the recommender which is $O(1)$ in complexity and weigh each of the received recommendation trusts to calculate the indirect trusts. Suppose there are n recommenders, the complexity of the feedback model is thus $O(n)$. The correlation module takes the direct's trust and indirect trust values as input to compute the dissimilarity test. Therefore, the correlation module requires $O(Nn)$ in complexity where N is the number of nodes in the network and n is the number of recommenders providing feedback. Next, the fusion module uses the Dempster's rule of combination to fuse two trust records together i.e. the direct trust and the indirect trust. The complexity is given by $O(2^S)$ where S is the number of elements in the frame of discernment and 2^S corresponds to the number of interactions of the two mass functions. When there are N nodes and n recommenders, the overall complexity of the fusion module is given by $O(Nn * 2^S)$.

With these values, the total runtime of the entire trust model is estimated as $O(N)+O(n)+O(Nn)+O(Nn*2^S)$ where the complexity is dominated by the fusion module. However, we only consider two elements in the frame of discernment. Hence, the complexity of the DS-Trust is low which is $O(4Nn) \approx O(Nn)$. Furthermore, the trust computations are carried out every trust monitoring period T which simplifies the complexity of the whole model as $O(Nn/T)$.

3.5 Conclusion

In this chapter, we have introduced two interesting properties of DST that makes it attractive for use in trust aggregation in trust systems. We went on to develop a trust model called the DS-Trust model that makes use of these two properties to mitigate reputation-based attacks such as badmouthing and ballot-stuffing attacks.

As described, our proposed DS-Trust model consists of five modules: a monitoring module, a feedback module, a correlation module, a fusion module and a decision module. The monitoring module monitors the next hop forwarding promiscuously and formulates the direct trust using the entropy function to describe the unreliability of promiscuous listening. The correlation module performs dissimilarity test between the direct trust and all the received recommendation trusts to determine the amount of conflict in the trust records. The fusion module then uses the results of the correlation module to re-evaluate the contribution of the indirect trust value before fusing the direct trust and indirect trust together using the Dempster's rule of combination.

We have demonstrated numerically that the DS-Trust model is capable of handling highly misleading trust information and can effectively resist the effects of badmouthing and ballot-stuffing attacks compared to the linear opinion pooling, subjective logic model, entropy-based probability model and regression analysis approaches. In addition, we have applied the DS-Trust model to two different V-Mesh network architectures and perform extensive NS-3 simulations to demonstrate that DS-Trust is resilient to packet dropping attacks and is able to recover from the blackhole and grayhole attacks. More specifically, DS-Trust is able to improve the PDR and the throughput of the network with an acceptable routing overhead.

In our simulation, we have used the randomwaypoint as the mobility which is

not realistic for V-Mesh network. As a future work, we would also like to extend our simulation to include more realistic mobility models such as the datasets from the TAPASCologne Project [120]. This will help us to understand the impact of real mobility on the trust computations and evaluations.

Chapter 4

Authenticated Trust Model with Reinforced Overhearing

In chapter 3, we describe the DS-Trust model that uses the overhearing technique to carry out trust modeling and trust evaluations. As discussed in section 2.3.1.1, overhearing is unreliable and is vulnerable to limited transmission power and packet modification attacks. In this chapter, we extend the DS-Trust model presented in chapter 3 by augmenting it with countermeasures to overcome these limitations. In particular, each node performs upstream monitoring to reinforce the results gathered from downstream monitoring to improve the accuracy of overhearing. Additionally, we provide authentication capabilities by incorporating the Merkle tree-based authentication mechanism in the trust model to detect packet modifications attacks. Furthermore, we exploit the mobility patterns of the vehicle to derive a weight for discounting the recommendations to further improve the trust evaluation. To the best of our knowledge, this is the first work that addresses the shortcomings of overhearing and data integrity as a whole. Our proposed model is called the MeTRO trust model.

The rest of this chapter is organized as follows. Section 4.1 introduces the Merkle tree concept and describes the construction of the vehicle's mobility profile. Section 4.2 describes the MeTRO model in details. Section 4.3 analyzes the security properties of the MeTRO trust model. Section 4.4 analyzes the efficiency of the Merkle tree-based authentication mechanism in our model. Section 4.5 highlights the performance improvements of the MeTRO trust model under various attacks. Finally, section 4.6 concludes this chapter.

4.1 Preliminaries

In this section, we describe the process of generating the Merkle commitment value, which is necessary for authenticating incoming data packets. We also describe the process for modeling the mobility pattern of a vehicle.

4.1.1 Message Commitment Value

To preserve the integrity of data packets along a multi-hop path, we propose an efficient authentication mechanism based on the principles of Merkle tree [121] which, in essence, is a binary tree built upon a one-way hash function $h(\cdot)$. Each interior node of a Merkle tree is the hash of the concatenation of its left and right hash values. With its tree-like structure, it provides an efficient and secure verification to ensure that a packet, which is a part of a larger message, is received without modification. However, we defer the discussions of the authentication process to section 4.2.2 and focus on the process of establishing the commitment value in this section.

Suppose a source vehicle has a variable-length message M_i to send to the destination node, it first fragments the message into $n+1$ data chunks as follows: $d[0], d[1], d[2], \cdots, d[n]$ to conform to the payload requirements of a vehicular message format. These fragmented packets or data chunks are buffered while waiting to be transmitted. Next, each data chunk is encrypted for confidentiality using the pairwise keys shared between the source and the destination node which can be generated using techniques presented in [99, 102, 91, 88, 96]. A one-way hash function $h(\cdot)$ then hashes each encrypted data as follows: $Enc\{d[i]\}$ for $i \in n$ into U_i such that $U_i = h(Enc\{d[i]\})$ as shown in Figure 4.1.

As shown in Figure 4.1, each node further up the tree is the hashes of its respective left and right children. For example, the value U_{12} is the result of hashing the concatenation output of U_1 and U_2 i.e. $U_{12} = h(U_1 \parallel U_2)$ where $\parallel$ denotes the concatenation. This hashing process continues until the commitment value ($U_{1234578}$) is obtained. After that, the source vehicle disseminates the commitment value securely to the other nodes via the route discovery process. Suppose the AODV routing protocol [118] is used to discover a route to a destination vehicle D, the RREQ packet broadcast by the source vehicle contains the following fields as shown in (4.1):

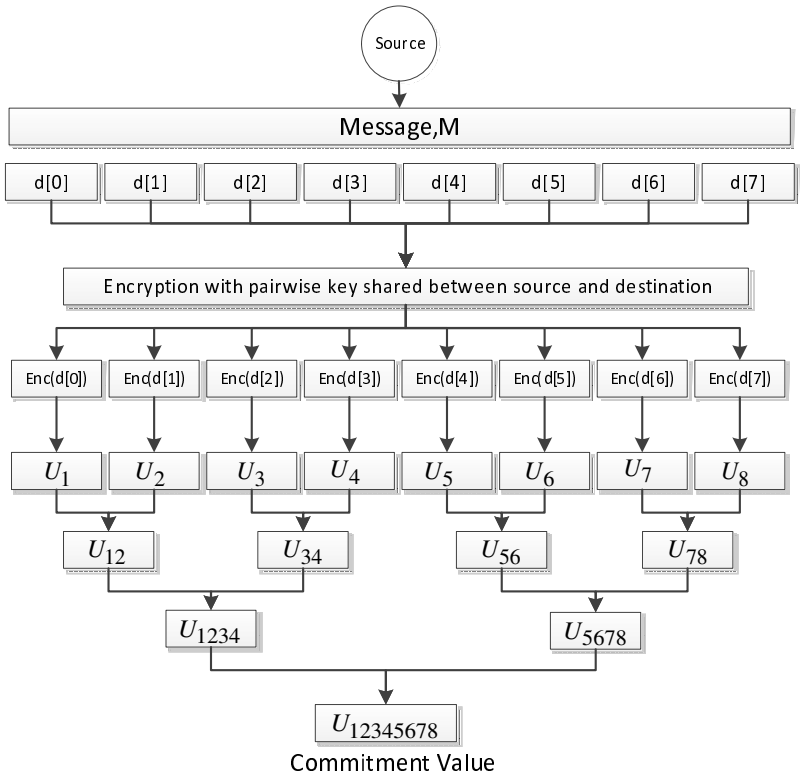

Figure 4.1: Merkle tree construction with 8 data blocks.

$$\begin{aligned} RREQ = ID_{RREQ} \parallel ID_S \parallel ID_D \parallel SEQ_S \parallel SEQ_D \\ \parallel CommitmentValue \parallel Sig_S \parallel Cert_S \end{aligned} \tag{4.1}$$

where ID_{RREQ} is the ID of the RREQ packet when taken in conjunction with the source node's IP address, ID_S is the ID of the source vehicle and ID_D is the ID of the destination vehicle. The SEQ_S and the SEQ_D are the source and destination sequence number respectively to ensure all routes are loop free. The commitment value field is a non-mutable field containing the expected root value of the Merkle tree i.e. $U_{12345678}$ according to Figure 4.1. Sig_S is the digital signature created by the source vehicle covering all the parameters to the left to prove authenticity and integrity of the message. $Cert_S$ is the digital certificate of the source vehicle to attest to the validity of the source's public key. When an intermediate node receives the RREQ packet, it checks if it has a path to the destination. If there is a valid route to the destination, the node unicasts a RREP back to the source vehicle and stores the commitment value for later use. In the event that there is a link breakage and a new route discovery is necessary, the source vehicle does not need to re-send the commitment value since all the nodes have already received it in the first round

of route discovery. A new commitment value only needs to be generated and shared publicly when the source has a new message M_{i+1} for the destination or when the last packet of the Merkle tree corresponding to the message M_i has been sent out. To differentiate the commitment values in the network, the source ID and the source sequence number, encapsulated in the RREQ packet can be used to identify each value uniquely and ensuring its freshness.

4.1.2 Mobility Profiling

The next requirement is to create a mobility profile for each vehicle. It is generated by means of tracking the vehicle's driving patterns and keeping state of the paths taken by the vehicle. Based on the mobility profile, we derive a weight to denote how often those paths are taken. A higher weight is associated with paths that are taken more often than the others. The derived weights are then used as a weighting function to discount the recommendation trusts during the trust evaluation process. As a general rule, the recommendation trust should carry more weight if it comes from vehicles that use the route frequently as compared to vehicles that travel on these paths less often. In this way, the effect of the recommendation trusts, especially from the unreliable sources, in particular, the badmouthing attackers is mitigated thereby producing more accurate trust evaluations.

To create the mobility profiles, we leverage on the beacon messages broadcast by the vehicles. As stated in [2, 31], a beacon message is disseminated periodically once every 300ms to all the neighbors over a single hop. It contains information like the velocity, position and time, which can be exploited by the RSU to configure the mobility profile. Suppose we have a scenario as given in Figure 4.2.

User A leaves home at 8am in the morning and drives to work along the route demarcated by the black color line. In the afternoon, User A drives out for lunch and goes back to the office where he continues his work until 6:30pm before returning home using the same route but in the opposite direction. As the vehicle travels along the paths, it broadcasts beacon messages indicating its geographical location as discussed before. Each RSU in the vicinity of the travel paths records the time as vehicle A passes through the area and store this information in the database, i.e. RSU 1 and RSU 7 stores $<$ Vehicle A, 8:15am, RSU1 $>$ and $<$ Vehicle A, 8:45am, RSU7 $>$ in the database respectively. After a pre-defined interval, the

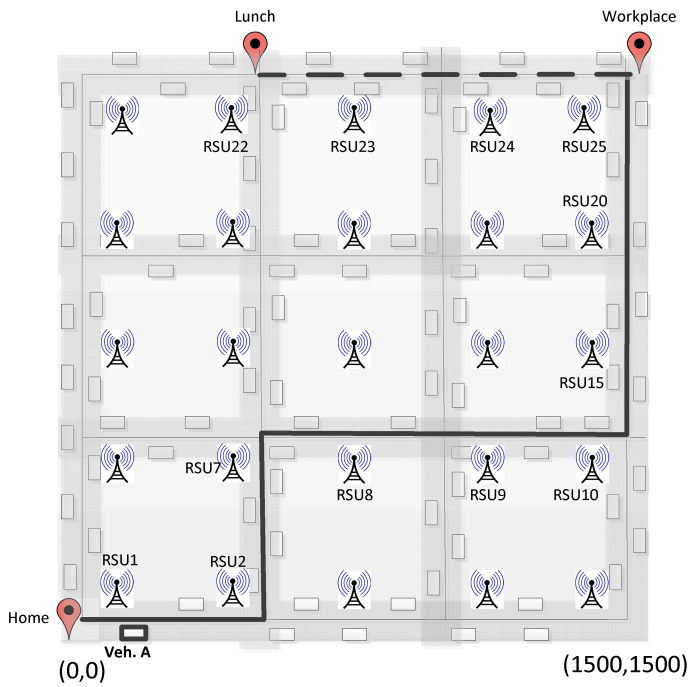

Figure 4.2: Mobility pattern of vehicle A.

RSUs exchange the shared information with one another and start constructing the paths taken by the vehicle during the interval.

Suppose each day is divided into 4 periods of 6 hours each, the mobility profile of vehicle A held by each RSU will contain the following information as given in Table 4.1. To allow for a finer granularity, the pre-defined interval can be made smaller to capture the dynamics of the driver's behaviors. The mobility profiles are then combined over a week where the same paths are added up cumulatively. By the same paths, we mean a path that contains the same set of RSU IDs in any order. For example, the forward paths and the reverse paths to and back from work are considered the same if both paths contain the same set of RSU IDs. After that, the weight of each path denoted by ω_i is calculated using (4.2). An example is provided in Table 4.2 with some arbitrary paths and data to illustrate the computation of the weights. In section 4.2.4, we will discuss how the weights are utilized for the trust computation when we describe the trust components of our trust model.

$$\omega_i = \frac{\text{number of times path i is taken}}{\text{total paths recorded per week}} \tag{4.2}$$

Table 4.1: Mobility profile of vehicle A in each RSU

Period	**Path**
0600-1200	RSU(1-2-7-8-9-10-15-20-25)
1200-1800	RSU(25-24-23-22)
1800-0000	RSU(25-20-15-10-9-8-7-2-1)
0000-0600	-

Table 4.2: Aggregated profile of vehicle A in each RSU

Path	**Count**	**Weight(ω)**
RSU(1-2-7-8-9-10-15-20-25)	14	0.5
RSU(25-24-23-22)	6	0.214
RSU(20-15-10-9)	8	0.286

However, we note that by exposing the location of the vehicle to the RSU, it may endanger the driver's privacy. Therefore, mobility profiling must incorporate some privacy-preserving techniques[1] as a future work.

4.2 MeTRO Trust Model

In this section, we describe the components of the MeTRO trust model. It consists of six modules as shown in Figure 4.3: the downstream module, the upstream module, the direct trust evaluation module, the recommendation module, the fusion module and the decision module. For ease of exposition, we refer to n_{i-1} and n_{i+1} as the precursor node and successor node of n_i respectively. Furthermore, we consider the scenario where n_{i-1} evaluates the trust of n_i when describing the trust components.

[1]More specifically, pseudonym-based schemes can be used, as described in [122]. The idea behind the pseudonym-based scheme is to replace the actual ID of the vehicle in the beacon message by an abstract identifier that changes after a certain period of time. In doing so, it limits the scope of the adversary through which the position of the vehicle can be traced back to the actual driver. Another method to preserve privacy is to use the group signature, where a signature is generated using the individual private keys of the vehicles, but it can be verified using a group-wide public key.

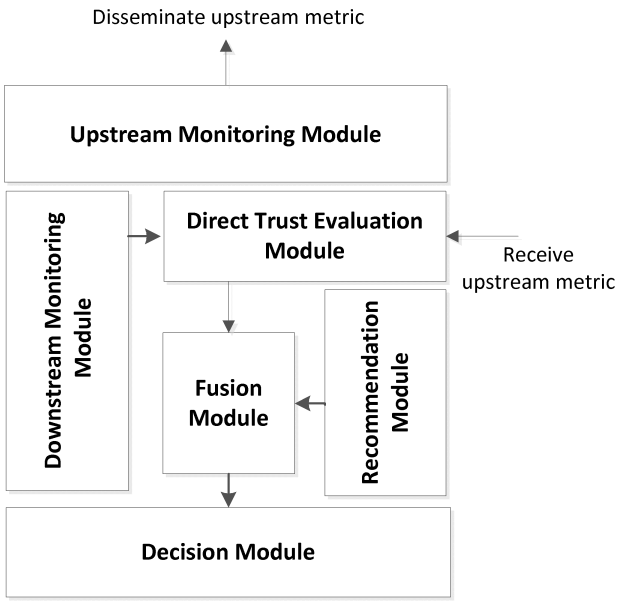

Figure 4.3: Proposed MeTRO trust model.

4.2.1 Downstream Monitoring Module

The downstream module is equipped with the overhearing functionality [36] to monitor the number of packets sent and the number of the packets overheard. More specifically, n_{i-1} maintains a list of sent packets by storing the packet IDs and the corresponding IDs of the next hop i.e. n_i. If n_{i-1} overhears a packet and found a match in the corresponding table, it implies that n_i has forwarded the data packet successfully. Consequently, a packet counter is incremented to denote a successful event. The whole process of checking and updating the packet counter takes place over a time window called the monitoring interval which we denote by τ. After τ interval has elapsed, n_{i-1} computes the downstream metric $O_i^{downstream}$ for n_i using (4.3) where n_o is the number of overheard packets in the packet counter and n_f is the total number of packets sent to n_i for forwarding. The downstream metric is then passed over to the direct trust evaluation module for further processing.

$$O_i^{downstream} = \frac{n_o}{n_f} \tag{4.3}$$

4.2.2 Upstream Monitoring Module

The function of the upstream module is to verify that the received data from n_i has not been modified during the packet transition. As such, the verification step is carried out by n_{i+1} instead of n_{i-1}. To validate the integrity of the data packet, the source node appends the Merkle authentication path along with the data chunk to be authenticated as shown in (4.4).

$$Message = Enc_{sk}\{d[i]\}, AuthPath \tag{4.4}$$

where $d[i]$ is encrypted using the ephemeral pairwise key sk shared between the source and the destination to protect the plaintext against any unauthorized disclosure. $AuthPath$ denotes the authenticated path containing a list of sibling hash values which n_{i+1} needs to process to authenticate $Enc_{sk}\{d[i]\}$. To summarize, n_{i+1} makes use of the authentication path and the commitment value that was disseminated earlier during the route discovery process to validate the packet's integrity. There is no need to encrypt or sign the authentication path since any changes to the authentication path would invalidate the received data and would be detected by n_{i+1} during verification.

As an example, suppose that the source node sends the authentication path $AuthPath = [U_4, U_{12}, U_{5678}]$ to authenticate the data packet $d[2]$ in Figure 4.1. In this case, n_{i+1} computes a series of hashes in the following order: $U_3 = h(Enc_{sk}\{d[2]\})$, $U_{34} = h(U_3 \parallel U_4)$, $U_{1234} = h(U_{12} \parallel U_{34})$ and lastly, $\bar{U}_{12345678} = h(U_{1234} \parallel U_{5678})$. If the computed commitment value $\bar{U}_{12345678}$ is the same as the expected value $U_{12345678}$ stored by n_{i+1}, it accepts the data $d[2]$ or the encrypted form of the message $Enc_{sk}\{d[2]\}$ as valid. If the verification is unsuccessful, it means that n_i has modified the data or the message is a replayed one. In short, the commitment value attests to the validity of the data that the source vehicle is going to transmit. If one data chunk is added/removed or modified, it changes the hash of its parent and the commitment hash value will be different. As long as there is an honest node along the routing path, any packet modifications can be detected efficiently.

To track the number of successfully authenticated packets, a packet counter is implemented in this module that increments on each correctly received packet. Another packet counter is also introduced to count the number of packets received for authentication. Based on these two values, n_{i+1} computes an upstream metric

$O_i^{upstream}$ for n_i every τ monitoring interval using equation (4.5).

$$O_i^{upstream} = \frac{n_s}{n_r} \tag{4.5}$$

where n_s refers to the number of successful authentications and n_r denotes the number of packets received from n_i for authentication. However, the upstream metric $O_i^{upstream}$ is not used by n_{i+1} locally. It is sent to n_{i-1} via the broadcast mode where it is received by the direct trust evaluation module of n_{i-1}. To prevent any broadcast storm, the TTL of the broadcast packet is set to 5.

4.2.3 Direct Trust Evaluation Module

The direct trust evaluation module is the core component of the MeTRO trust model. It evaluates the trust value of a node based on the locally derived downstream metric and the upstream metric gathered from n_{i+1}. In our model, the direct trust value of a node n_i is given by (4.6).

$$DT_i = W_i \frac{n_s}{n_f} \tag{4.6}$$

where DT_i denotes the direct trust of node i evaluated locally, W_i denotes a weight that determines how much the ratio term to the right of it contributes towards the trust value, n_s refers to the number of successful authentications, and n_f refers to the total number of packets sent to n_i for forwarding. To determine the weight in (4.6), we use the Jensen Shannon Divergence (JSD) [123] to measure the closeness between the downstream metric and the upstream metric and subsequently, convert the result of JSD into an appropriate weight using (4.7).

$$\begin{aligned} W_i &= [1 - JSD(O_i^{downstream}, O_i^{upstream})] \\ &= [1 - H(\alpha_1 O_i^{downstream} + \alpha_2 O_i^{upstream}) - \alpha_1 H(O_i^{downstream}) - \alpha_2 H(O_i^{upstream})] \end{aligned} \tag{4.7}$$

where H is the Shannon entropy function expressed as $H(p) = -plog_2 p - (1 - p)log_2(1-p)$ in logarithmic base 2 and α_1,α_2 are the weights given to the downstream metric $O_i^{downstream}$ and the upstream metric $O_i^{upstream}$ respectively.

When calculating the JSD of the downstream and upstream metrics, α_1 can be selected such that $\alpha_1 > \alpha_2$ to indicate that the $O_i^{downstream}$ metric carries more

weight than the $O_i^{upstream}$ metric. Alternatively, the weights can be selected arbitrary as long as the following conditions are satisfied: α_1, $\alpha_2 \geq 0$ and $\alpha_1 + \alpha_2 = 1$. The value of the JSD is always non-negative and is bounded by 1 i.e. $0 \leq JSD(O_i^{downstream}, O_i^{upstream}) \leq 1$ when logarithmic base 2 is assumed. If the JSD value is close to 0, it implies that the downstream metric and upstream metric are almost similar. If the JSD has a value of 1, it denotes that the downstream metric and upstream metric are in total disagreement.

Based on these properties, we configure the weight such that W_i is large for small values of JSD and small when the JSD value between the upstream and downstream metrics is large. In other words, the direct trust value of a node is influenced by how much the downstream metric and the upstream metric supports each other. Once the direct trust value is obtained, it is passed to the fusion module.

4.2.4 Recommendation Module

Besides evaluating the trust value locally, the MeTRO trust model relies on the recommendation trusts issued by the neighboring nodes. As pointed out by [50, 51, 52, 53], gathering recommendation trusts improves the detection time of selfish nodes as opposed to evaluating trust based only on direct observations.

In our model, the recommendation module is triggered by the fusion module when it receives a request to evaluate the trust value of n_i from the direct trust evaluation module. Consequently, the recommendation module sends out requests to solicit feedbacks about n_i from its one-hop neighbors. Any neighbors within the radio range of n_{i-1} checks its own record to see if there is a trust record for the target node n_i. If a record is found, the neighboring nodes send their trust opinions about n_i to n_{i-1}. These trust opinions that originate from the recommenders are direct trust values of n_i calculated the same way as described in the direct trust evaluation module. To account for the reliability of the received recommendation trusts, n_{i-1} weighs the recommendation trust values to form the indirect trust value of n_i using equation (4.8).

$$IDT_i = \omega DT_i \tag{4.8}$$

where IDT_i is the indirect trust value of n_i after passing the recommendation trusts DT_i through a weighting function and ω is a weighting factor extracted from the

mobility profile of the recommender. The weighting factor ω is used to discount the recommendation trusts as described in section 4.1.2. It is obtained separately via a request message to a nearby RSU.

When the RSU receives the request message, it takes note of the current traveling path of the recommending vehicle and checks the past week mobility profile of the recommender. If there is a match, the RSU returns the weight ω that corresponds to the current path taken by the recommending vehicle. In this case, vehicles traveling on paths that are taken very often based on the past week's mobility profile would have their recommendation trust emphasized by a heavier weight ω. After the recommendation trusts are weighted, they are forwarded to the fusion module where they are combined with the direct trust value.

4.2.5 Fusion Module

The role of the fusion module is to combine the direct trust and the indirect trust together to form an overall trust value for n_i. If the indirect trust value is not available, the fusion module outputs the direct trust directly to the decision module, otherwise we adopt the concept of DST [113] for combining the direct and indirect trust values. The idea of trust fusion is the same as the approach presented in the DS-Trust model. Thus, we direct readers to section 3.2.4 for implementation details. Nevertheless, for the sake of completeness, we also discuss the important equations here.

First, we define a dissimilarity ratio for checking the similarities between the direct trust and indirect trust values as defined in (4.9). The results are used to determine the contribution of the indirect trust prior to aggregation in DST.

$$dissim(DT_i, IDT_i) = \frac{|DT_i - IDT_i|}{|DT_i| + |IDT_i|} \tag{4.9}$$

where $dissim(DT_i, IDT_i)$ denotes the variance between the two trust records with DT_i denoting the direct trust value and IDT_i representing the indirect trust respectively.

Based on the results of the dissimilarity test, a set of classification rules is formulated in (4.10) and (4.11) which assigns the $dissim(DT_i, IDT_i)$ value as the belief mass for uncertainty whereas the belief mass for the rest of the subsets i.e. $m(T)$ and $m(\bar{T})$ are updated such that $m(T) + m(\bar{T}) + m(T, \bar{T}) = 1$.

If IDT $\geq \gamma$, then

$$\begin{aligned} m(T,\bar{T}) &= dissim(DT_i, IDT_i) \\ m(T) &= 1 - m(T,\bar{T}) \\ m(\bar{T}) &= 0 \end{aligned} \tag{4.10}$$

Else

$$\begin{aligned} m(T,\bar{T}) &= dissim(DT_i, IDT_i) \\ m(\bar{T}) &= 1 - m(T,\bar{T}) \\ m(T) &= 0 \end{aligned} \tag{4.11}$$

As for the classification of direct trust value, the rule is that, if the direct trust value obtained from (4.6) is larger or equal to the detection threshold γ, the node n_i will be classified as belonging to the Trusted (T) set, otherwise it is classified as an Untrusted ($\bar{T}$) set as given by the following classification rules defined in (4.12) and (4.13).

If DT $\geq \gamma$, then

$$\begin{aligned} m(T,\bar{T}) &= 1 - DT \\ m(T) &= DT \\ m(\bar{T}) &= 0 \end{aligned} \tag{4.12}$$

Else

$$\begin{aligned} m(T,\bar{T}) &= 1 - DT \\ m(\bar{T}) &= DT \\ m(T) &= 0 \end{aligned} \tag{4.13}$$

Once the indirect trust value and the direct trust value have been classified properly, they are fused together using the Dempster's rule of combination to form an aggregated belief as shown in (4.14).

$$m_{i,j}(s) = \frac{1}{1-K} \sum_{p \cap q = s} m_i(p) m_j(q) \quad s \neq \phi \tag{4.14}$$

$$K = \sum_{p \cap q = \phi} m_i(p) m_j(q) \tag{4.15}$$

where $m_{i,j}(s)$ denotes the combined belief mass for proposition s, $m_i(p)$ and $m_j(q)$ represent the belief mass assigned by evidence source i and j respectively. The quantity K represents the belief mass associated with conflict and serves as a normalization factor to ensure that the total sum of the combined masses, $m_{i,j} = 1$. If more than one sources of evidences is available, they can be combined in a pairwise manner using equation (4.16).

$$m_{DS}(s) = m_1(s) \oplus m_2(s) \oplus \cdots \oplus m_J(s) \quad \forall s \in 2^{\Theta} \tag{4.16}$$

where $m_{DS}(s)$ denotes the resulting mass function after combination, $\oplus$ represents the Dempster's rule of combination as described in (4.14) and (4.15) and all the masses of the form $m.(s)$ denotes the belief masses assigned by evidence sources $\{1, 2, 3, \cdots, J\}$. Once the overall trust value is determined, it is then sent to the decision module for further actions.

4.2.6 Decision Module

If recommendation trusts are not available, the decision module will base its decision only on the direct trust value obtained from the direct trust evaluation module. Otherwise, the combined trust value as calculated per the fusion module is used. Below summarizes the actions taken by n_{i-1} when the trust value of n_i falls below the detection threshold γ.

- **Isolate selfish nodes** - n_{i-1} blacklists the misbehaved node i.e. n_i based on a back-off algorithm. When a node is detected as misbehaving for the first time, it is put on a blacklist for a period of time called the back-off time δ. After δ has expired, the misbehaved node is removed from the blacklist and re-joins the network where it can participate in the communication again. However, if the misbehavior is detected for the second time, the back-off time will increase exponentially and accordingly with respect to the number of misbehavior detection. This relationship can be described by (4.17).

$$\delta_{current} = \delta^i \tag{4.17}$$

  where $\delta_{current}$ is the current back-off time and i is the number of times the node is detected for misbehavior. Besides blacklisting the misbehaved node, n_{i-1}

also conducts a blacklist broadcast throughout the network to inform other nodes to isolate it from all future communications.

- **New route discovery** - When a node is blacklisted, n_{i-1} will send a RERR message back to the source node to initiate a new route discovery to find a path free of misbehaved nodes.

4.3 Security Discussion

In this section, we analyze the security properties of the MeTRO trust model. In particular, we describe how the proposed model can resist eavesdropping attacks, message replay attacks, packet modification attacks and non-repudiation attacks.

4.3.1 Eavesdropping Attacks

The main goal of eavesdropping attacks is to gain access of confidential data to know more about the network such as the users' transactions, the capabilities of the other road users on the network and the types of resources available. The attackers can then use this information to launch other attacks to bring down a particular network service. The MeTRO trust model is robust against eavesdroppers because the message or data chunks are encrypted for confidentiality using pairwise keys that are unique only to the source and the destination vehicles. Furthermore, the derived pairwise keys are kept in the HSM for extra protection against physical tampering.

Without the knowledge of the pairwise keys, it is difficult for any vehicles in the network to deduce the plaintext from the ciphertext. Furthermore, by encrypting the message, our model also guarantees data privacy which is not only critical in the V-Mesh, but it also influences the penetration rate of many other vehicular related applications.

4.3.2 Message Replay Attacks

Replay attack is a type of active attack where a compromised attacker captures a valid copy of the message and replays it at a later time for personal gains. An example of such attack is to repeat packets sent by the ambulance to obtain a faster ride. However, replay attacks are not possible in our model because messages are

encrypted using the ephemeral pairwise keys which are generated at random per session.

Let us consider the scenario that a compromised attacker captures an encrypted message of the form $Enc_{sk'}\{d[i]\}$ where $Enc_{sk'}\{d[i]\} \neq Enc_{sk}\{d[i]\}$ due to $sk' \neq sk$ and replays it to the destination or any other nodes along the routing path for forwarding to the destination. Using the replayed encrypted message $Enc_{sk'}\{d[i]\}$ and the given authentication path, the recipient computes the commitment value.

In this case, the computed commitment value will not match the current root value stored since the ephemeral keys have expired and are no longer valid for the current session. Furthermore, if the computed commitment value is found in the list of commitment values that have been authenticated before, the message is likely to be a replayed one and hence, will be discarded.

4.3.3 Packet Modification Attacks

The packet modification attack is another type of active attacks which, when left undetected, are damaging to the vehicular communication networks because it influences the behavior of other road users. Our model, however, remains secure under the packet modification attacks because it uses the Merkle-based tree authentication mechanism to verify the integrity of the messages.

Here, we assume the worst case scenario where the compromised attacker has acquired the pairwise encryption key for the current session and uses it to encrypt the modified message of its own where $d'[i] \neq d[i]$. Therefore, the goal of the attacker is to find a message $d'[i]$ such that it produces the same hash value denoted by $h(Enc_{sk}\{d'[i]\}) = h(Enc_{sk}\{d[i]\})$.

In our model, this is computationally infeasible since we adopt a cryptographic hash function with a 256-bit output. Using the birthday paradox, we can deduce that it would take approximately $1.17(2^{128})$ attempts to generate a collision with a probability of 0.5. Similarly, it is difficult to change the authentication path to make the commitment value match.

4.3.4 Repudiation Attacks

Repudiation attacks can happen when the sender denies having sent the message. This is not possible in our model because it makes use of the Merkle tree concept. In our model, the sender is required to compute a Merkle commitment value which is disseminated to the road users before the actual message chunks are forwarded. The commitment value serves to record the chronology of each data's existence so that subsequent nodes along the forwarding path can verify the integrity of each received data block. The verification is carried out by checking if the computed commitment value matches the given value. If the verification is successful, it means that the message originates from the sender, hence, proving non-repudiation property. Moreover, the commitment value is authenticated using digital signature in (4.1) which further protects it against any tampering.

4.4 Efficiency Analysis

In section 4.2.2, we mentioned that our model is able to detect modified packets as they are propagated along a routing path. This is achieved by integrating the Merkle-based tree authentication mechanism into our model. In this section, we focus on investigating the authentication performance of our model by measuring the time needed to generate the commitment value and the time required to authenticate a data packet. Since the ECDSA [29] was proposed as the de facto standard to secure vehicular communications, we further compare the authentication delay of our model with the ECDSA to demonstrate its efficiency improvements under increasing messaging load.

4.4.1 Generation Time of Commitment Value

To evaluate the root generation time, we implemented the Merkle algorithm in C++ and simulated the program for various tree heights ranging from 1 to 10. In our implementation, we assume a complete binary tree where each interior node has two child nodes. Therefore, the height H of the Merkle tree is given by $log_2 n$ where n is the number of data blocks after message fragmentation. We evaluate the performance for two scenarios where a variable-length message is fragmented into

blocks size of 200 bytes and 512 bytes respectively. Furthermore, we use the SHA-256 as the hashing algorithm to provide 128 bits of security against the collision attacks.

Simulations are conducted on a 32 bit Ubuntu operating system with 1GB RAM running on an Intel Core i7-2620M@2.70GHz workstation. The results are shown in Figure 4.4 and Figure 4.5, where the line graph represents the amount of time required to generate the commitment value and the bar graph represents the delay of authenticating a data block, respectively. Each data point on the plots is calculated based on an average of 100000 simulation runs. Here, we note that the generation of the commitment values is performed only by the source node while the authentication of a data block is performed by the intermediate hops along the routing path. These two timings are plotted to understand how the source and the intermediate nodes perform relative to each other.

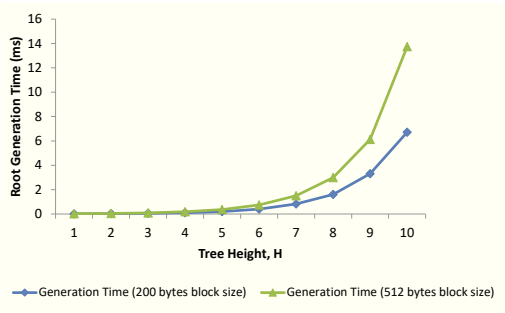

Figure 4.4: Delay for generating the Merkle root.

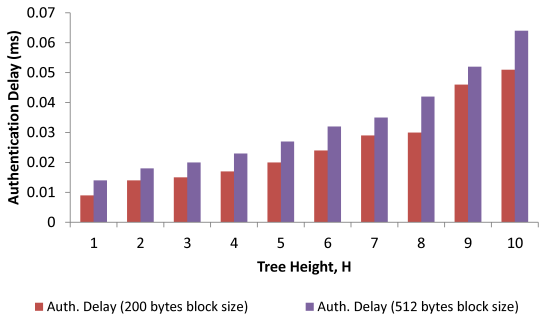

Figure 4.5: Delay for authenticating a data block.

Comparing Figure 4.4 and Figure 4.5, we observe that the generation time of a commitment value is much higher than the authentication delay of a data block. This is because the source node needs to hash n data blocks at the base of the tree followed by $(n-1)$ hash computations at the interior nodes of the tree in order to calculate the commitment value. On the other hand, each intermediate node only requires $log_2(n)$ hashing operations to validate a data block since one sibling hash value at each level of the tree is provided in the authentication path. The results also show an increase in the generation time and the authentication delay when a message is fragmented into larger blocks. This increase is due to the hashing of a

larger block size.

Furthermore, as the tree depth increases each round, both the delay times start to increase exponentially without bounds. This result suggests that the cost of constructing a Merkle tree over a large file is going to scale up exponentially, which is not practical for real time applications. To keep the computation low, we propose to construct multiple Merkle trees over a large file by splitting it into more manageable fragments. This means that a larger file would now have more than one commitment values for verification.

4.4.2 Authentication Delay

Next, we evaluate the authentication delay performance of our model when a node is subject to an increasing number of messages and compare our results to the ECDSA [29]. In this experiment, we assume the height of the tree is 10. Consequently, based on the results in Figure 4.5, the authentication delay of verifying a data block of size 512 bytes and 200 bytes is 0.064ms and 0.051ms, respectively. To capture the verification delay of the ECDSA signature, we implement the ECDSA algorithm in C language using the Openssl package with a key length of 256 bits.

Simulations are carried out on a 32-bit Debian Linux operating system with 2GB RAM running on an Intel Core i7-2620M@2.7GHz laptop. The average delay based on 100000 simulation runs is evaluated at 2.832ms. Using these values, we evaluate numerically the total authentication delay experienced by a node by both models when the number of messages received changes from 50 to 300. The delay performance is illustrated in Figure 4.6 where the left inset figure is a blow-up of our model based on the different block sizes.

Result shows that the authentication delay increases linearly with the number of messages received. However, our model is still far more superior when compared to the ECDSA algorithm in the main figure. The delay of our model as seen in Figure 4.6 is several orders of magnitude lower than the delay incurred by the ECDSA which requires 141ms to authenticate 50 messages. With a lower authentication cost, it means that our model is more scalable than the ECDSA as it is able to support more messages in the network. Secondly, a lower authentication delay makes our model very efficient for many delay intolerant applications which is critical in a V-Mesh network.

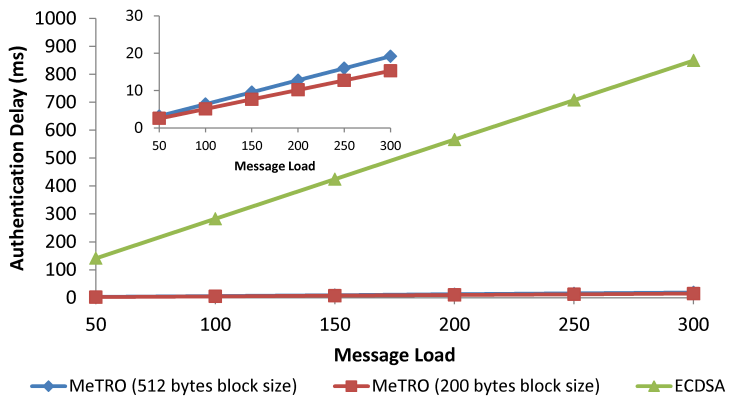

Figure 4.6: Authentication delay of different schemes as a function of message load.

4.5 Performance Evaluation

Here, we focus on the performance analysis of the MeTRO trust model under the effects of limited transmission power attacks, packet modification attacks and packet dropping attacks. To this end, the MeTRO trust model is integrated into the AODV protocol which we denote by MeTRO-I and simulated using NS-3 simulator [119]. We also simulate a variant of our model without the upstream monitoring as a comparison model to highlight the performance gains of using upstream observation. We denote this comparison model as MeTRO-II. We further compare our model with the baseline AODV without any trust mechanism to demonstrate the benefits of employing a trust model.

4.5.1 Simulation Setup

The simulation topology is a 3x3 Manhattan Grid model to mimic an urban V-Meshs scenario as shown in Figure 4.2. The dimension of the simulation environment is 1.5km by 1.5km with a grid spacing of 500m. There are 75 mobiles vehicles deployed in the simulation environment, traveling at an average speed of 15m/s or equivalently about 54km/h. Furthermore, the moving vehicles are configured to pause for 5s with a pausing probability of 0.5 to model traffic lights conditions on the roads. The vehicle movements are generated using the Bonnmotion tool [124]. In addition, 25 static RSUs are deployed near road junctions and they are connected among themselves to form a mesh network to bridge communication dead zone. The transmission range of the vehicle and the RSU is 500m. The rest of the simulation

parameters are shown in Table 4.3 and we define the following performance metrics for comparison purposes:

Table 4.3: Simulation parameters

Simulation tool	NS-3
Grid spacing	500m
Transmission range	500 m
Network area	1500 m x 1500 m
Packet size	512 Bytes
Packet generation rate	4 packets/s
Simulation time	300 s
Traffic type	CBR
Transport protocol	UDP
Mac protocol	IEEE 802.11p
Propagation loss model	RangePropagationLossModel
Physical layer	YansWifiPhy channel
Detection Threshold, γ	0.5
No. of nodes in the network	75 mobile vehicles, 25 static RSUs
Traffic	10 source-destination pairs
Routing Protocol	AODV

- PDR refers to the ratio of the number of delivered packets to the number of packets generated by the CBR sources.
- Modified packets ratio refers to the number of modified packets detected over the total number of messages sent.
- NRO refers to the number of routing control packets and trust related control packets transmitted per data packet delivered at the destination.
- End-to-End (E2E) delay refers to the average time taken by a packet to arrive at a destination from the source. It includes delays during route discovery and delays at the interface queues.

4.5.2 Performance under Limited Transmission Power Attacks

Figure 4.7 shows the PDR performance of the three schemes when the number of limited transmission power attackers is varied from 5 to 30 in the simulation. It is observed that the MeTRO-I model has the highest PDR. The PDR improvement is about 11% and 5% compared to that of the baseline AODV and the MeTRO-II model respectively. The improvement is due to the introduction of a weighting factor W in the trust formula as defined in (4.6) which serves to discount the observations based on the closeness between the upstream and downstream observations.

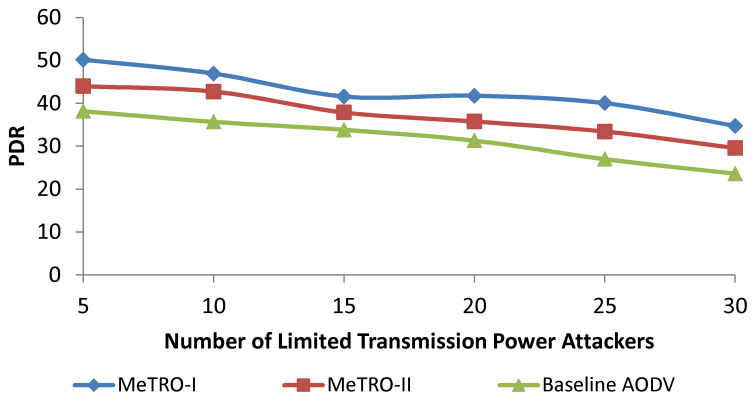

Figure 4.7: PDR of different schemes under limited transmission power attacks.

Suppose the attacker conducts limited transmission power attacks with a given probability distribution, the upstream observations as perceived by the successor node on the attacker will be very small due to the non-receipt of data packets whereas the precursor node perceives the attacker as behaving normally. Due to the conflicting values, there will be a large discrepancy in the two observations which results in a smaller weighting factor and a larger discount on the contribution of the forwarding term $\frac{n_s}{n_f}$. And, if the derived trust value falls below the detection threshold γ, the node will be isolated and a new route will be selected around these attackers, thereby alleviating the effects of limited transmission power attacks. In contrast, the MeTRO-II is basically an overhearing scheme capable of only downstream monitoring. Consequently, the precursor node always detects the attacker as good since the packets are overheard correctly. Due to this reason, the MeTRO-II

model experiences random packet loss contributing to a smaller PDR improvement and a lower PDR compared to the baseline AODV and our model.

4.5.3 Performance under Packet Modification Attacks

Next, we investigate the performance of MeTRO-I model when the network is under packet modification attacks. In this simulation, we count the number of modified packets detected and expressed this value as a ratio over the total number of packets sent. Figure 4.8 illustrates the performance of the various schemes for different number of modification attackers. We observe that MeTRO-I has the lowest fraction of modified packets among the three schemes. The percentage of modified packets in our model is approximately 20% and 30% lower than that of the baseline AODV and MeTRO-II respectively.

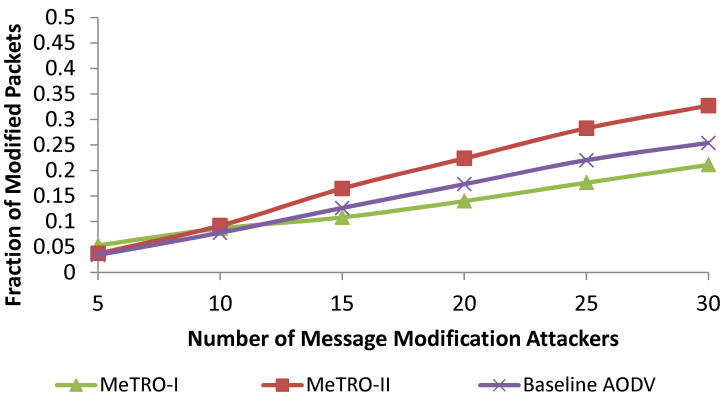

Figure 4.8: PDR of different schemes under packet modification attacks.

Our model has a better performance because each received data packet is authenticated using the Merkle-based tree authentication mechanism. The number of correctly authenticated packets is further tracked as part of the upstream monitoring process carried out by the successor node which is then feedback to the precursor node for computing the trust value. If the attacker modifies the data packets, the number of successfully authenticated packets in the forwarding term $\frac{n_s}{n_f}$ defined in (4.6) will be very small and hence, results in a smaller trust value. If the computed trust value falls below the pre-defined threshold γ, a new route discovery will be initiated to avoid these packet modifying attackers.

The MeTRO-II model, on the other hand, has the highest fraction of modified

packets despite claims that overhearing can be used to detect modified packets [62]. One reason is due to the missed overhearing caused by inevitable network collisions. Even if overhearing is assumed perfect, a large bulk of the modified packets has already been forwarded by the next hop and left to propagate throughout the network which contributes to a higher modified packets ratio. In comparison, our model drops the modified packets when they are detected followed by updating the precursor node with the most current upstream observations so that packet modifying attackers can be detected and avoided early.

4.5.4 Performance under Packet Dropping Attacks

In terms of packet dropping attacks, we consider two types, namely the blackhole attacks and the grayhole attacks. In the first experiment, we vary the number of packet droppers in the network from 5 to 30 and configure them to drop 100% of the recieved packets to simulate the blackhole attacks. In the second experiment, we fix the number of packet droppers in the network as 20 but configure them to drop packets selectively at different rates ranging from 10% to 100%. For both experiments, the performance is evaluated in terms of the PDR, E2E and NRO performance.

4.5.4.1 Resiliency to Blackhole Attacks

Figure 4.9 compares the PDR performance of the various schemes when the network is under blackhole attacks. We observe that the PDR performance of the MeTRO-I model is comparable to the MeTRO-II model. Both models are able to detect blackhole attackers and they produce better PDR performance than the baseline AODV. For both MeTRO models, the average PDR is about 40%, while that of the baseline AODV is about 20%. The PDR is improved because of the ability to find an alternate path around the misbehaved nodes to transmit the rest of the packets to the destination.

Figure 4.10 illustrates the E2E delay performance. Simulation results show that the MeTRO-I model has the lowest E2E delay compared to the baseline AODV and the MeTRO-II models. The E2E delay of the MeTRO-I model remains consistent at 0.2ms throughout the entire range of simulated blackhole vehicles because it relies on information from both the upstream and downstream monitoring which

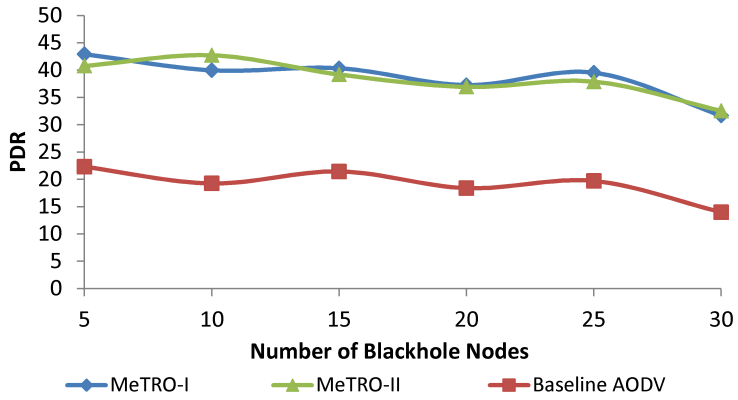

Figure 4.9: PDR of different schemes as a function of blackhole vehicles.

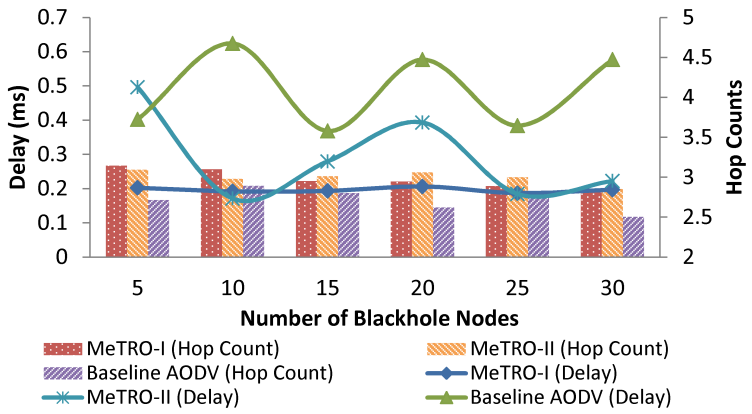

Figure 4.10: E2E delay of different schemes as a function of blackhole vehicle.

enables it to select neighbors that are more reliable in carrying out forwarding. As a result, the MeTRO-I model produces more stable network connectivity with less re-transmission. On the other hand, the baseline AODV and the MeTRO-II models are observed to have erratic delay timings with many fluctuations as they suffer from frequent routing breaks that require re-initialization of routing path. Since both MeTRO models establish a routing path around malicious nodes, both of them experience a higher hop count than the baseline AODV as seen in the Figure 4.10.

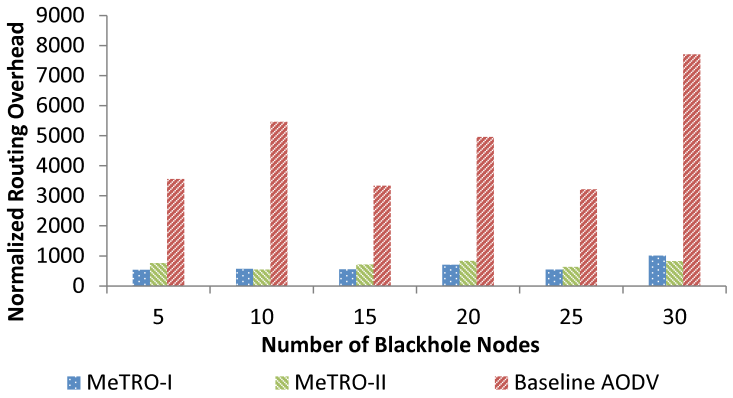

Figure 4.11: NRO of different schemes as a function of blackhole vehicles.

Next, we compare the NRO performance of the various schemes. As seen in Figure 4.11, the NRO performance is consistently low for the MeTRO-I model and the MeTRO-II model in comparison to the baseline AODV. This result highlights the efficiency and scalability of our model in coping with increasing blackhole vehicles. The baseline AODV, on the other hand, has the highest NRO due to more routing breaks and frequent re-initialization of paths.

4.5.4.2 Resiliency to Grayhole Attacks

For the case of grayhole attacks, we observe a similar trend in the PDR, E2E and NRO performance. For instance, the PDR performance in Figure 4.12 demonstrates that both the MeTRO-I and MeTRO-II models can improve the PDR to about 40% despite packets are being dropped selectively at various different rates. The PDR improvement is almost 20% compared to the baseline AODV model without any trust model employed. In terms of the E2E performance, results in Figure 4.13 show that MeTRO-I model has the lowest delay performance despite an increased

in the number of hops to reach the destination. In comparison, the baseline AODV has the highest delay performance because of the constant route discovery to locate more reliable routes. Lastly, the NRO performance results in Figure 4.14 indicate that both the MeTRO models are very efficient as they do not introduce much routing overhead.

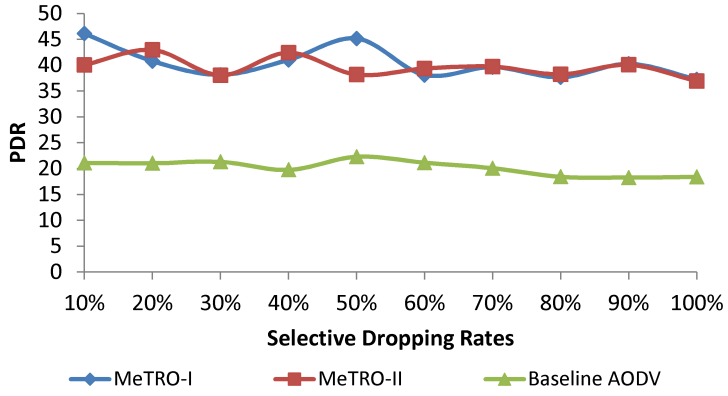

Figure 4.12: PDR of different schemes under different dropping rates.

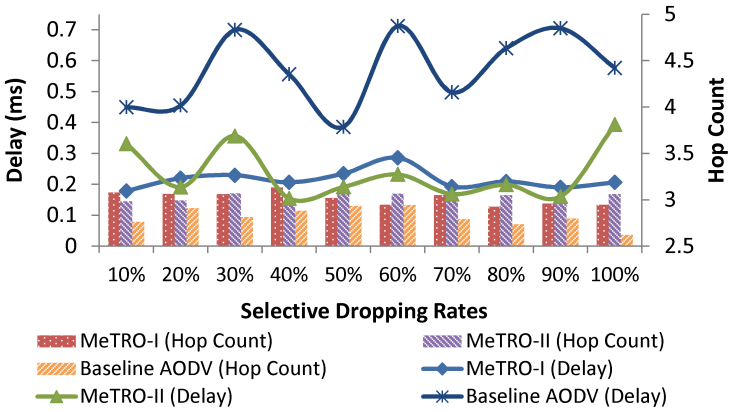

Figure 4.13: E2E delay of different schemes under different dropping rates.

We note here that though the MeTRO-II model is equally effective as the MeTRO-I model in detecting blackhole and grayhole attacks, the MeTRO-I model is still far more superior than the MeTRO-II model because it is able to defend against the limited transmission power attack and packet modification attacks. This highlights the novelty and effectiveness of our model.

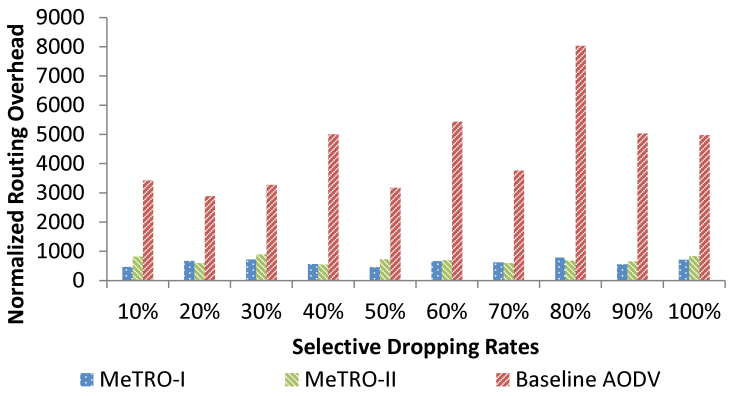

Figure 4.14: NRO of different schemes under different dropping rates.

4.6 Conclusion

In this chapter, we have proposed the MeTRO model to overcome the overhearing problem. Our trust model makes use of both upstream monitoring and downstream monitoring to track a node's forwarding behavior. Based on these two observations, a combined metric is derived which is then used as a weight to express the trust value of nodes for improving the trust evaluation.

Besides protecting the network against packet droppers, it is also important to secure the network against packet modifying attackers, especially in a V-Mesh network since message authenticity and integrity is critical to the proper functioning of the network. Therefore, we integrate an efficient authentication scheme based on the Merkle Tree concept into our model to enable the intermediate nodes along a forwarding path to validate received packets.

Extensive performance evaluation shows that our MeTRO is secure against limited transmission power attacks, packet modification attacks and common packet dropping attacks. Moreover, we have shown that MeTRO is very efficient in terms of the authentication delay and the generation time when compared to the de facto ECDSA standard. We have also provided informal security discussions to prove that our model can resist eavesdropping, replay, modifications and repudiation attacks.

In this work, the privacy concern is the location tracking because the RSUs are assumed to be aware of the movement patterns of the vehicles. As part of our future work, we plan to investigate these issues in greater depth and then extend our model to deal with such attacks. Moreover, we plan to consider collusion attacks where

two or more nodes along a path collude with one another. For example, one of the nodes will modify the packet contents while the downstream node will always report good behavior on its upstream node during the upstream monitoring process. We plan to study this attack scenario in greater detail and propose a solution to counter such attack.

Chapter 5

Secure and Authenticated Key Management Protocol

In this chapter, we propose a SA-KMP framework to reduce the dependence on expensive PKI operations. More specifically, our work is a combination of two concepts namely, the PKR scheme [104] and the 3D grid-based key distribution scheme [37, 38]. The objective of the PKR scheme is to facilitate the distribution of up-to-date and certified public keys so that resource-intensive CRL management can be eliminated. On the other hand, the 3D grid-key distribution scheme is used to negotiate a set of keys for use by symmetric encryption to avoid the expensive operations associated with asymmetric cryptography. We further introduce several extensions to the PKR and 3D grid-key distribution schemes to defend against the DoS attackers and node capture attacks.

The rest of this chapter is organized as follows. Section 5.1 states our design goals followed by a discussion of the enhancements made towards the two schemes. Section 5.2 describes the components of the SA-KMP scheme. Section 5.3 provides a formal and informal security validation of our scheme. Section 5.4 analyzes the efficiency and scalability of SA-KMP. Finally, section 5.5 concludes this chapter.

5.1 Introduction

As mentioned in chapter 1, a V-Mesh network is characterized by short contact times and intermittent connectivity due to the high mobility of vehicles. Therefore, our primary objective is to develop a scheme that can facilitate the establishment of

secure communications seamlessly and efficiently. To achieve this objective, we have identified 6 sub-goals which can be broken down into two categories namely, performance and security goals. These goals are critical to the development of SA-KMP as they form the basis of our design. First, we elaborate on our goals. Then, we describe how the PKR and the 3D grid-key distribution schemes are enhanced to combat DoS and node capture attacks.

Performance Goals

1. ***Efficient and light weight*** - SA-KMP has to be efficient due to the strict delay requirements of most vehicular applications. Distribution and verification of public key certificates must not be too complex or time-consuming that results in a high overhead and delay [40]. At the same time, it should not compromise on the level of security. It also has to be light-weight as the vehicles have limited computing capabilities in terms of CPU power compared to the RSU.

2. ***Scalability and adaptability*** - With the increase in the number of vehicles in the network, the number of certificates or keys that are issued or revoked will also increase. Therefore, it is important that SA-KMP is able to adapt to the growth of the network and scales well.

Security Goals

1. ***Message integrity*** - All messages in a V2I or V2V communication and vice-versa need to be protected against any data modification to ensure integrity. If messages are modified intentionally, it could lead to adverse consequences such as injures or loss of lives.

2. ***Message authenticity*** - Recipients of a message must also be able to verify that the message is authentic and is sent by a user who possesses a valid signing key. This often has to be carried out on every new message.

3. ***Message non-repudiation*** - Message sent or received must be able to guarantee non-repudiation property, that is, the sender or receiver of a message cannot later deny having sent or received the message. Ways to achieve non-repudiation are to use digital signatures, timestamps and etc.

4. ***Message confidentiality*** - Message must always be kept secret and protected from unauthorized third party or eavesdroppers at all time. This property can be easily realized using proper encryption technique.

5.1.1 Enhancements to PKR

5.1.1.1 DoS Mitigation

As highlighted in Section 2.4.2.1, a malicious vehicle may launch DoS attacks by sending many invalid public key request messages to the RSUs to bring down the service. To prevent DoS attacks, Wasef et al. [125] have proposed to append a HMAC to all the outgoing signed messages when the ratio of the invalid signatures to the total number of received messages in a given time period exceeds a certain threshold. When the other vehicles receive the message, each of them calculates the HMAC on the received message and compares the calculated hash value with the received HMAC. If the value matches, the vehicle proceeds to verify the signature of the message. Otherwise, the vehicle drops the message immediately. One problem with this approach is that it requires several invalid signatures to trigger the threshold in order to detect the presence of DoS attackers.

Different from [125], we adopt a more proactive approach whereby two communicating parties take turns to verify each other signatures before issuing the requested public keys. Since the verification process requires both parties to expend an equal amount of CPU resources, there is no real incentive for the vehicle to launch DoS attacks. External DoS attackers are also prevented since any vehicles without valid public/private keys will fail the authentication process and be excluded from the communications. In order to ease the computational burden on users performing the authentication, we adopt the Schnorr signature scheme [109].

In terms of security, the Schnorr signature scheme is based on the hardness of computing discrete logarithms and the infeasibility of inverting a cryptographic one-way hash function [109]. It is provably secure in the random oracle model [126] and has the following three properties: completeness, soundness and zero-knowledge.

- Completeness ensures that a prover who knows the discrete logarithm is always able to pass the verification challenge.

- Soundness ensures that an adversary who does not know the discrete logarithm has only a negligible probability to pass the verification challenge.
- Zero-knowledge property proves that a prover leaks no more than one bit information to the verifier regardless of whether the prover knows the discrete logarithm.

Moreover, it is lightweight and efficient because it requires only one modular exponential, one modular addition and one modular multiplication to generate the signature. On the other hand, Digital Signature Algorithm (DSA) requires an additional modular inverse operation which is considered the most expensive part of the signature generation process. We illustrate the difference between the Schnorr signature and the DSA in Figure 5.1.

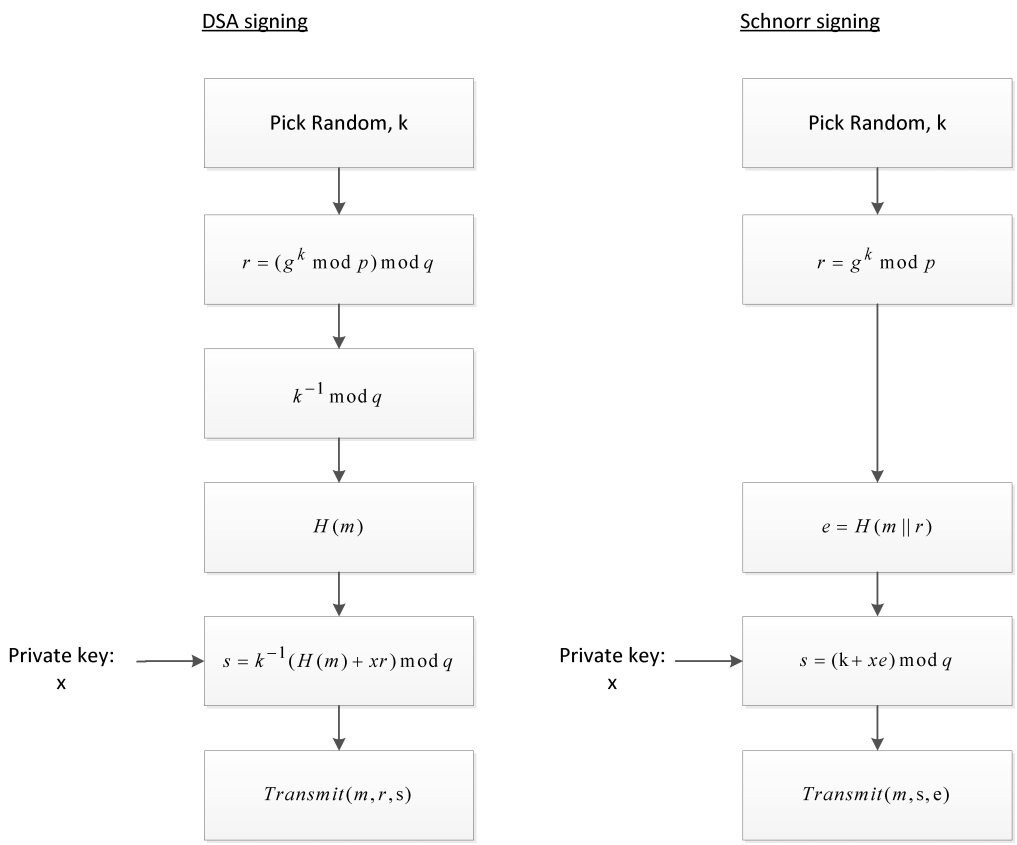

Figure 5.1: Difference between Schnorr and DSA signing process

To make our scheme even more time-efficient, we use the ECC technology [127] due to its shorter key size requirement and faster speed compared to the non-elliptic

curve equivalents. Correspondingly, our authentication scheme is called the EC-Schnorr and the security is based on the Elliptic Curve Discrete Logarithm Problem (ECDLP). We conclude the discussion of the DoS mitigation technique by emphasizing that the Schnorr signature scheme is designed to mitigate DoS attacks from internal vehicles rather than external attackers. We do not consider the case where an external attacker may exploit the properties of the Schnorr signature to exhaust the resources of a hacked vehicle. We also do not consider the fact that an external attacker can launch a Distributed Denial of Service (DDoS) attack on the RSU by compromising multiple good vehicles simultaneously to send invalid request messages. To counter these attacks from an external attacker, we direct readers to Section 6.2.2 where we discuss several extensions to enhance the SA-KMP scheme.

5.1.1.2 Collusion Mitigation

Another issue of the PKR scheme [104] is that the RSUs are preloaded with the TA's signing key. Therefore, the malicious RSUs may also misbehave or collude with others to forge a valid public key reply message containing an incorrect public key. In our design, we propose not to issue the TA's signing key to the RSUs. In this way, the risks of forgery and collusion attacks are minimized. However, we still need to ensure the authenticity of the public keys in the directory. To do that, we propose that the TA creates a digital signature over the vehicle's ID and its corresponding public key to certify its validity. This generation of signature is carried out during the registration phase. The generated signature is then stored alongside the tuple <vehicle's ID, public key> in the directory. When the RSU receives a public key request message, it simply retrieves the TA's signature in the directory and append it to the public key reply message back to the requestor. The requestor then verifies the signature using the TA's public key to verify the correctness of the issued public key.

5.1.1.3 Distribution of Public Keys

In addition to the above enhancements, we also propose that the TA creates another public key directory and distributes it to all the vehicles to facilitate communications. To distinguish the two directories, we denote the public key directory that is assigned to the vehicles as the RSU Public File Directory (RSU-PFD) whereas

the directory that is kept by the RSUs as the Vehicle Public File Directory (VPFD). The rationale of having two different public key directories is to reduce the storage overhead requirements. That is, the RSU-PFD directory should only contain the RSU's ID and its associated public key while the VPFD directory only stores the vehicles' ID and their associated public keys. By giving each vehicle and RSU a copy of the directory, V2I communications can commerce immediately without the need to request for public keys. This eliminates unnecessary latency and overheads associated with the checking of certificates and downloading of CRLs. When a new vehicle joins the network, only the VPFD needs to be updated which is done in real time. After that, the updated copy of the VPFD is disseminated to all the RSUs via a secure network. In contrast, the RSU-PFD rarely gets updated because the number of RSUs is usually fixed. However if there is a need to, the vehicles can still receive their updated copy of the RSU repository from the Operator Equipment Manufacturers (OEMs) as part of the vehicle periodic maintenance.

5.1.2 Enhancements to 3D Grid-based Key Distribution

5.1.2.1 Node Capture Attacks Mitigation

The downside of the 3D grid-key distribution scheme [37, 38] is that keys are generated beforehand and preloaded onto each vehicle and RSU prior to deployment. As such, this approach is vulnerable to node capture attacks which can compromise the entire key space. A workaround solution is to update the key space periodically. However, the update frequency has not been discussed and studied. If the update period is too short, it will incur a high communication overhead since extra messages have to be sent to revoke the current set of keys and to re-distribute the new set of keys. If the update period is too long, recovery from attacks if any, will be too late.

To solve this problem, we propose to generate the keys dynamically instead of preloading them. To achieve this, each communicating party plays a part in choosing a random nonce value for generating the common keys. After that, the two random nonce values are hashed together repeatedly. The number of times to hash will depend on the solution points (i, j, k) of the equation groups. Since the common keys are generated on demand and are guaranteed unique due to the random nonce values chosen by the users, our proposed scheme is protected against

node capture attacks. Even if a node is captured by an adversary, only the affected node's communication is compromised. The rest of the nodes in the network can still continue communications. Consequently, huge communication costs associated with the updating of the key space are avoided. More details of the key derivation process will be presented in Section 5.2.4.

5.1.2.2 DoS Mitigation

We also wish to highlight that the key distribution schemes that we have reviewed before in Section 2.4.1.1 have not addressed the DoS attacks properly. Since key request messages usually contain information to establish a key, they need to be digitally signed for authenticity and integrity. Due to this, a vehicle or RSU can send many forged messages with invalid signatures forcing the receiving parties to perform unnecessary signature verifications. To secure the 3D grid-based key distribution scheme against the DoS attacks, we augment the scheme with an EC-Schnorr based authentication scheme as described in the previous section. Similarly, each participating vehicle or RSU must prove to each other the intent to derive the common keys by committing some of their CPU resources. More details of the key agreement scheme will be discussed in Section 5.2.3

5.2 SA-KMP Scheme

The SA-KMP scheme consists of five modules as shown in Figure 5.2. The five modules are the System Setup, Public Key Request, Key Agreement, Key Derivation and the Revocation modules. In this section, we provide details of these modules and provide examples to illustrate the establishment of pairwise keys and group keys.

5.2.1 System Setup Module

This module is installed in the TA and consists of 3 steps.

Step 1 (Initialization): The deployment area is divided into a 3D space ($m \times m \times m$) where m represents the size of an area. Each entity occupies a unique GPS location in the 3D space denoted by (i, j, k). The TA first determines the domain

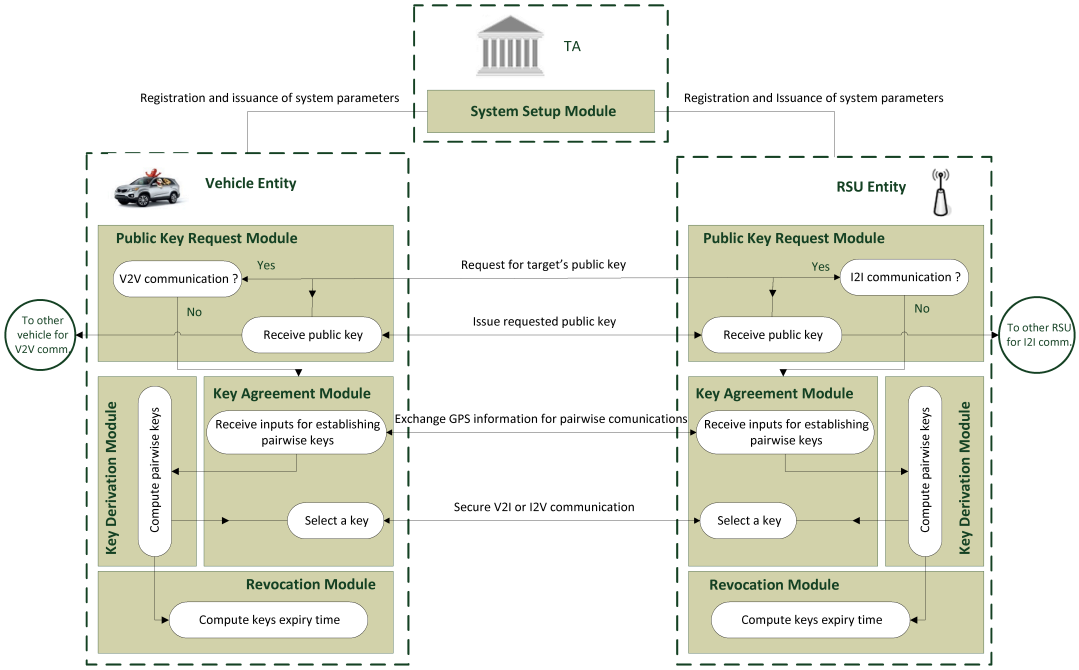

Figure 5.2: SA-KMP framework.

parameters (p, a, b, G, n, h) for an elliptic curve E over a finite field F_p where p is a large prime number, a and b are the coefficients of the elliptic curve $E : y^2 \,(mod\, p) = x^3 + ax + b\,(mod\, p)$, G is a generator represented by a point (G_x, G_y) on the elliptic curve, n is the order of the generator, and $h : (0,1)^* \rightarrow F_p$ is a secure hash function. After defining the ECC domain parameters, TA publishes the domain parameters (p, a, b, G, n, h) including its public key denoted by k^+_{TA} to all the entities in the network while keeping its own private key k^-_{TA} secret. Next, TA generates a set of N plane equations defined by (5.1)

$$z - k_i + c_1(y - j_i) + c_2(x - i_i) = 0\, mod(m) \tag{5.1}$$

The number of plane equations that needs to be generated by the TA must be at least 3 i.e. $N \geq 3$ so that we can solve for the three unknowns (x, y, z). For $N > 3$, any group of 3 distinct equations can be chosen to generate one solution. In other words, given $N \geq 3$ linear equations, there will be $N(N-1)(N-2)$ sets of equations to produce $N(N-1)(N-2)$ solutions. The (i_i, j_i, k_i) coordinate in each plane equation denotes the GPS location of entity i where i represents the entity's ID. The parameter, m denotes the size of the 3D space and c_1, c_2 represents the coefficients of a plane equation. The coefficients are uniquely chosen by TA such that $c_1 \neq c_2$.

This ensures that the solution to a system of three plane equations is always unique. Next, the TA generates another three hash functions (H_1, H_2, H_3) which are used for deriving the common keys. At the same time, the TA creates two directories, namely the RSU-PFD and the VPFD to store the entities' public keys.

Step 2 (Registration): Next, each entity i creates its own private/public (k_i^-/k_i^+) key pair and registers its public key with the TA. For each registrant, the TA generates a unique ID and binds the entity's public key to its ID. If the registrant is an RSU, the TA stores the binding $< ID, public\,key >$ inside the RSU-PFD as shown in Table 5.1. On the other hand, if the registrant is a vehicle, the TA stores the entry inside the VPFD. In addition to storing the bindings $< ID, public\,key >$, the VPFD also stores for each entry the TA's signature that is created from the vehicle's ID and its certified public key as shown in Table 5.2. This is to prevent a colluding RSU from issuing a wrong public key.

Table 5.1: Contents of RSU-PFD

RSU ID	**Public Key**
5	k_5^+
4	k_4^+
100	k_{100}^+
$\vdots$	$\vdots$

Table 5.2: Contents of VPFD

Vehicle ID	**Public Key**	**ID and Public key binding signed by TA**
1	k_1^+	$Sig_{k_{TA}^-}\{1, k_1^+\}$
2	k_2^+	$Sig_{k_{TA}^-}\{2, k_2^+\}$
9	k_9^+	$Sig_{k_{TA}^-}\{9, k_9^+\}$
$\vdots$	$\vdots$	$\vdots$

Step 3 (Dissemination): After successful registration, the TA disseminates a

read-only copy of the VPFD to each RSU and a read-only copy of the RSU-PFD to each vehicle. Both directories are updated whenever a new entity joins the network or when an entity's public key is revoked. The size of the VPFD is finite as a vehicle will be de-registered and removed from the directory upon reaching its end-of-life. The RSU-PFD rarely increases in size because the number of RSUs in the network is fixed. The benefits of disseminating up-to-date directories are threefold: (1) eliminates the need to download huge CRL, (2) reduces the time and complexity required to verify the certificates and (3) improve the timeliness and freshness of public keys. Next, the TA issues the N plane equations and the hash functions (H_1, H_2, H_3) to each entity which are stored in the HSM for protection.

5.2.2 Public Key Request Module

This module is installed in each entity. It is invoked when the vehicle or the RSU receives public key request messages from the other entities. It contains an authentication mechanism to mitigate the effects of internal and external DoS attacks. The key idea is to make each other verify an EC-Schnorr signature and proving their identities and authenticity at the same time. Figure 5.3 illustrates the necessary steps involved between a vehicle $\mathcal{V}$ and an RSU $\mathcal{R}$ when the public key request module is executed.

Step 1 and Step 2: First, $\mathcal{V}$ and $\mathcal{R}$ select one random value, $c_i \in [1, n-1]$ and a 128-bits random nonce R_i where i stands for the ID of the entity and computes C_i according to the formula in step 2 of Figure 5.3. The $R_{\mathcal{V}}$ and $R_{\mathcal{R}}$ values are nonce values to prevent the replay attacks while the $C_{\mathcal{V}}$ and $C_{\mathcal{R}}$ are commitment values used to ensure the entity authenticity and as a knowledge proof that the prover has verified the digital signature. In this case, both the nonce and commitment values are pre-computed off-line or during the idle state of the entity's processor to make the scheme efficient.

Step 3: Suppose $\mathcal{V}$ wants to communicate with another vehicle, it constructs a public key query message $M_{\mathcal{V}}$ containing its own ID and the ID of the target vehicle, e.g. ID_3 and sends it to $\mathcal{R}$. This message also contains the $\mathcal{V}$'s random nonce value $R_{\mathcal{V}}$.

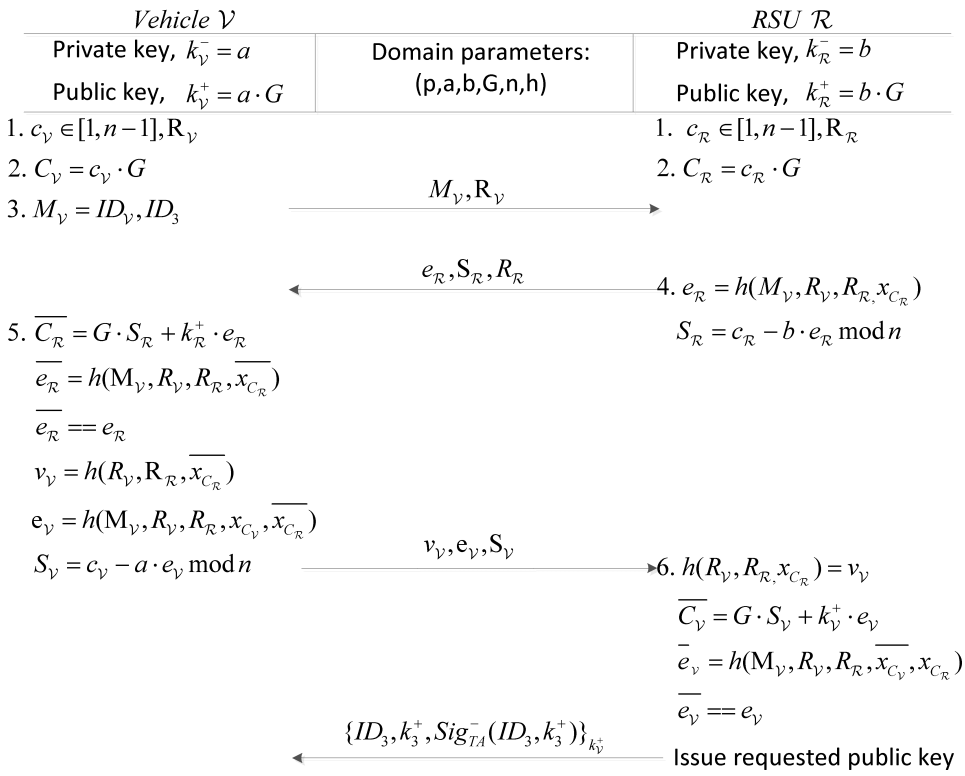

Figure 5.3: Authentication process in the public key request module.

Step 4: Upon receiving $(M_{\mathcal{V}}, R_{\mathcal{V}})$ from $\mathcal{V}$, $\mathcal{R}$ generates a signature pair $(e_{\mathcal{R}}, S_{\mathcal{R}})$ consisting of a challenge, $e_{\mathcal{R}}$ and a response, $S_{\mathcal{R}}$ according to the formula given by step 4 of Figure 5.3 and sends the tuple $(e_{\mathcal{R}}, S_{\mathcal{R}}, R_{\mathcal{R}})$ to $\mathcal{V}$. The challenge $e_{\mathcal{R}}$ contains the hash of the message, including the random nonce values $(R_{\mathcal{V}}, R_{\mathcal{R}})$ and the $x_{C_{\mathcal{R}}}$ coordinate of the commitment value, $C_{\mathcal{R}}$ of $\mathcal{R}$ that is used as a proof to determine if $\mathcal{V}$ has verified the signature.

Step 5: When $\mathcal{V}$ receives $(e_{\mathcal{R}}, S_{\mathcal{R}}, R_{\mathcal{R}})$ from $\mathcal{R}$, it retrieves the $\mathcal{R}$'s public key $k_{\mathcal{R}}^{+}$ from the RSU-PFD. Then, $\mathcal{V}$ re-computes the $\mathcal{R}$'s commitment value as $\overline{C_{\mathcal{R}}} = G \cdot S_{\mathcal{R}} + k_{\mathcal{R}}^{+} \cdot e_{\mathcal{R}}$ using the received values $S_{\mathcal{R}}$, $e_{\mathcal{R}}$ and $k_{\mathcal{R}}^{+}$. Next, $\mathcal{V}$ calculates its version of the challenge denoted as $\overline{e_{\mathcal{R}}}$ by taking the hash of the message, $M_{\mathcal{V}}$ concatenated with both the random nonce values $(R_{\mathcal{V}}, R_{\mathcal{R}})$ and the calculated $x_{\overline{C_{\mathcal{R}}}}$ coordinate of the $\overline{C_{\mathcal{R}}}$ computed earlier. If the calculated hash value matches the received $e_{\mathcal{R}}$, it implies that $\mathcal{R}$ is legitimate. Then, $\mathcal{V}$ proceeds to generate its own signature pair $(e_{\mathcal{V}}, S_{\mathcal{V}})$ followed by a proof denoted as $v_{\mathcal{V}}$ and sends the message tuple $(v_{\mathcal{V}}, e_{\mathcal{V}}, S_{\mathcal{V}})$ to $\mathcal{R}$ to complete the authentication process. The proof $v_{\mathcal{V}}$ is used

to verify if $\mathcal{V}$ has verified the signature of $\mathcal{R}$ and is calculated by hashing the two random nonce values and the $x_{\overline{C_\mathcal{R}}}$ coordinate of $\overline{C_\mathcal{R}}$ together. However, if $\mathcal{R}$ fails authentication, i.e. $\overline{e_\mathcal{R}} \neq e_\mathcal{R}$, $\mathcal{V}$ discards all message from $\mathcal{R}$ and terminates the session. The proof of correctness where $\mathcal{V}$ validates the signature of $\mathcal{R}$ is given by (5.2) and (5.3).

$$\begin{aligned}\overline{C_\mathcal{R}} &= G \cdot S_\mathcal{R} + k_\mathcal{R}^+ \cdot e_\mathcal{R} \\ &= G \cdot (c_\mathcal{R} - b \cdot e_\mathcal{R}) + k_\mathcal{R}^+ \cdot e_\mathcal{R} \\ &= G \cdot c_\mathcal{R} - k_\mathcal{R}^+ \cdot e_\mathcal{R} + k_\mathcal{R}^+ \cdot e_\mathcal{R} \\ &= c_\mathcal{R} \cdot G\end{aligned} \tag{5.2}$$

$$\overline{e_\mathcal{R}} = h(M_\mathcal{V}, R_\mathcal{V}, R_\mathcal{R}, x_{\overline{c_\mathcal{R}}}) \tag{5.3}$$

Step 6: Before $\mathcal{R}$ proceeds to verify the signature pair $(e_\mathcal{V}, S_\mathcal{V})$ of $\mathcal{V}$. It first checks that the hash value $v_\mathcal{V}$ is correct by computing $h(R_\mathcal{V}, R_\mathcal{R}, x_{C_\mathcal{R}})$ using its own commitment value, $x_{C_\mathcal{R}}$ of $C_\mathcal{R}$ generated in step 2. The signature pair $(e_\mathcal{V}, S_\mathcal{V})$ needs to be verified only if the proof matches. If the proof provided by $\mathcal{V}$ is incorrect, it implies that $\mathcal{V}$ has not verified the signature in step 5. Therefore, $\mathcal{R}$ discards the message without verifying. If the hash value $v_\mathcal{V}$ is correct, $\mathcal{R}$ proceeds to locate the public key of $\mathcal{V}$ in the VPFD and performs the verification process to compute the commitment value of $\mathcal{V}$ using $\overline{C_\mathcal{V}} = G \cdot S_\mathcal{V} + k_\mathcal{V}^+ \cdot e_\mathcal{V}$. Lastly, $\mathcal{R}$ verifies if $\overline{e_\mathcal{V}} = h(m, R_\mathcal{V}, R_\mathcal{R}, x_{\overline{c_\mathcal{V}}}, C_\mathcal{R})$ matches the received $e_\mathcal{V}$. If $\overline{e_\mathcal{V}} = e_\mathcal{V}$, $\mathcal{R}$ issues the requested public key. Otherwise, $\mathcal{R}$ terminates at this step. The reply message to $\mathcal{V}$ contains the requested ID, the requested public key and the TA's signature on the $< ID, public\,key >$ binding which are retrieved from the VPFD. The entire message is encrypted using $\mathcal{V}$'s public key to ensure confidentiality. The inclusion of the TA's signature is to ensure the integrity and authenticity of the message. When $\mathcal{V}$ receives the reply message, it decrypts the message using its own public key $k_\mathcal{V}^+$ and verify the TA's signature using k_{TA}^+. If the signature is correct, it means that the requested public key issued by $\mathcal{R}$ is correct.

5.2.3 Key Agreement Module

This module is in charge of negotiating keying material for establishing pairwise keys and group keys. Since it involves exchanging of signed messages, malicious attackers can send invalid messages to deplete the resources of the recipients leading to DoS attacks. To solve this, we incorporate the same authentication mechanism described in the public key request module where communicating parties must prove to each other their intentions by committing some of their CPU resources. However, the difference is that the exchanged messages now contain additional information such as the randomly generated nonce and the GPS information of $\mathcal{R}$ and $\mathcal{V}$. This information is required for establishing keys and is protected using public key cryptography. First, we present the protocols to establish the pairwise keys. After that, we discuss the procedure to establish group keys for group communications.

5.2.3.1 Pairwise Keys Agreement

Figure 5.4 describes the steps involved in establishing pairwise keys between $\mathcal{V}$ and $\mathcal{R}$ in a V2I communication.

Message 1: First, $\mathcal{V}$ initiates a request message to establish pairwise keys with $\mathcal{R}$. This request message consists of a random 128-bit nonce $R_{\mathcal{V}}$, the identifier of $\mathcal{V}$ ($ID_{\mathcal{V}}$) and the identifier of $\mathcal{R}$ ($ID_{\mathcal{R}}$). In this message, the nonce value $R_{\mathcal{V}}$ is added to avoid replay attacks. As this is only a request message, it does not need to be encrypted and digitally signed.

Message 2: When $\mathcal{R}$ receives the key request message, it generates a commitment value $C_{\mathcal{R}} = c_{\mathcal{R}} \cdot G$ where $c_{\mathcal{R}} \in [1, n-1]$. The commitment value $C_{\mathcal{R}}$ is used for two purposes. Firstly, it is used to verify the authenticity of $\mathcal{R}$ i.e. to prove that $\mathcal{R}$ has the knowledge of its own private key. Secondly, it is used by $\mathcal{R}$ to prove if $\mathcal{V}$ has verified the given signature before commencing the key agreement process. Aside from the commitment value, $\mathcal{R}$ also provides the location information $LOCM_{\mathcal{R}}$ which is basically the GPS location and a 128-bits nonce $N_{\mathcal{R}}$ for deriving the pairwise keys. In short, $\mathcal{R}$'s reply message $M_{\mathcal{R}}$ to $\mathcal{V}$ contains the following information:

$$(ID_{\mathcal{R}}, ID_{\mathcal{V}}, LOCM_{\mathcal{R}}, N_{\mathcal{R}})_{k^{+}_{\mathcal{V}}}$$

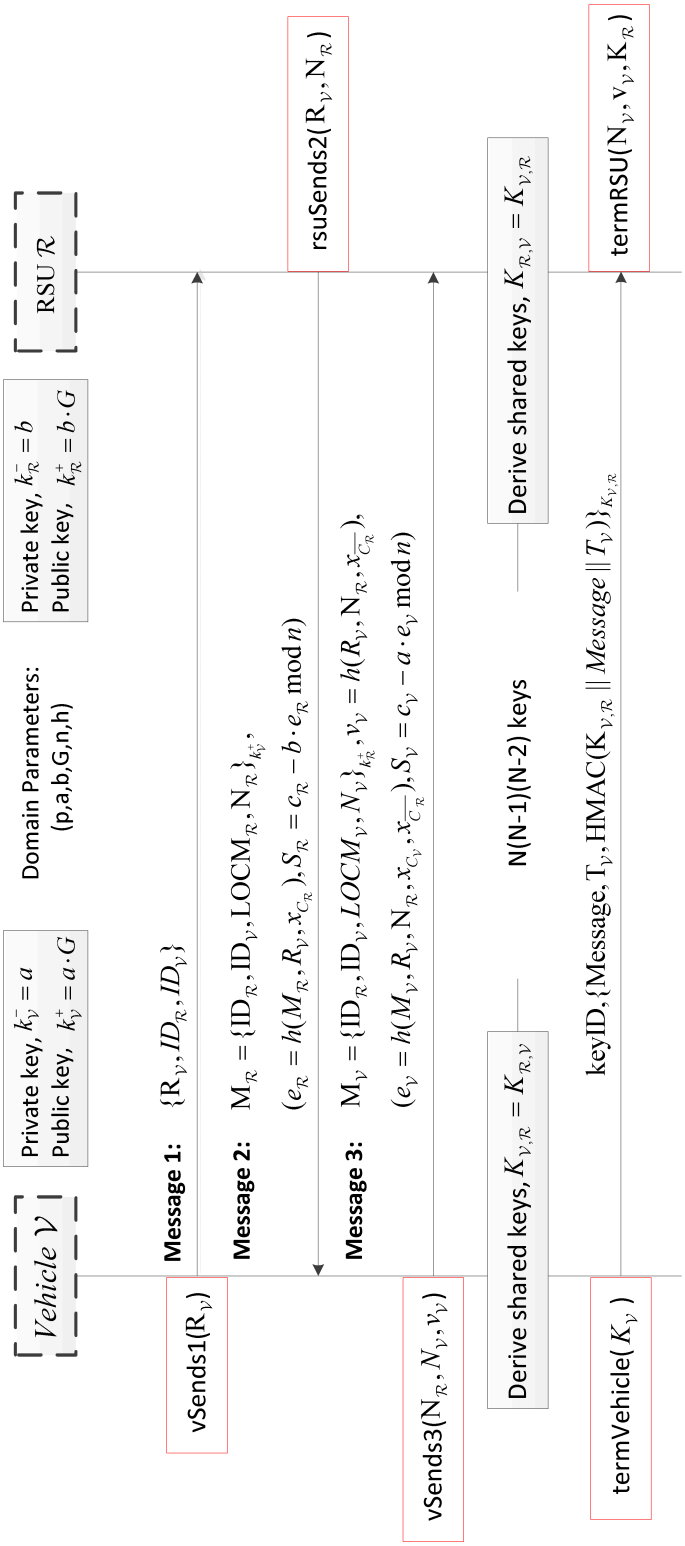

Figure 5.4: Exchange of keying materials for establishing pairwise keys in V2I.

All these information are encrypted using $\mathcal{V}$'s public key for confidentiality which is available in the VPFD maintained by each RSU. After that, $\mathcal{R}$ generates an EC-Schnorr signature denoted as $(e_{\mathcal{R}}, S_{\mathcal{R}})$ for $\mathcal{V}$ to verify. The challenge $e_{\mathcal{R}}$ generated by $\mathcal{R}$ is simply a hash of the message $M_{\mathcal{R}}$ concatenated with $\mathcal{V}$'s random nonce $R_{\mathcal{V}}$ and the $x_{C_{\mathcal{R}}}$ coordinate of the commitment value $C_{\mathcal{R}}$ generated by $\mathcal{R}$. The response $S_{\mathcal{R}}$ of the signature pair $(e_{\mathcal{R}}, S_{\mathcal{V}})$ is calculated the same way as per the public key request module described earlier and is given by $S_{\mathcal{R}} = c_{\mathcal{V}} - b \cdot e_{\mathcal{V}} \, (mod\, n)$.

Upon receiving the signature pair from $\mathcal{R}$, $\mathcal{V}$ first decrypts the message using its own private key k_1^- to retrieve $\mathcal{R}$'s GPS information $LOCM_{\mathcal{R}}$ and the nonce value $N_{\mathcal{R}}$ for calculating the pairwise keys. After that, $\mathcal{V}$ looks up $\mathcal{R}$'s public key $k_{\mathcal{R}}^+$ in the RSU-PFD and verifies the signature by computing its own version of the $\mathcal{R}$'s commitment value denoted as $\overline{C_{\mathcal{R}}}$ using $e_{\mathcal{R}}, S_{\mathcal{R}}$ and $k_{\mathcal{R}}^+$. Next, $\mathcal{V}$ calculates the hash of the message $M_{\mathcal{R}}$ concatenated with its own random nonce $R_{\mathcal{V}}$ and the $x_{\overline{C_{\mathcal{R}}}}$ coordinate of the commitment value, $\overline{C_{\mathcal{R}}}$ calculated earlier. If $\overline{e_{\mathcal{R}}} = e_{\mathcal{R}}$ given by $\mathcal{R}$, it implies that $\mathcal{R}$ is authenticated and is now $\mathcal{V}$'s turn to generate the information and the signature to establish the pairwise keys with RSU.

Message 3: After $\mathcal{R}$ has been authenticated, $\mathcal{V}$ creates a reply message $M_{\mathcal{V}}$ to $\mathcal{R}$. It contains the identities of $\mathcal{R}$ $(ID_{\mathcal{R}})$ and $\mathcal{V}$ $(ID_{\mathcal{V}})$ including the GPS location of $\mathcal{V}$ $LOCM_{\mathcal{V}}$ and a random nonce $N_{\mathcal{V}}$. The entire message $M_{\mathcal{V}}$ is encrypted using $\mathcal{R}$'s public key for confidentiality which is available in the RSU-PFD maintained by $\mathcal{V}$. Next, the proof $v_{\mathcal{V}}$ is sent to $\mathcal{R}$ to prove that it has indeed verified the signature. It is represented by the hash function of the random nonce of $\mathcal{V}$ and $\mathcal{R}$ $(R_{\mathcal{V}}, R_{\mathcal{R}})$ concatenated with the $x_{\overline{C_{\mathcal{R}}}}$ coordinate of the commitment value of $\mathcal{R}$. After that, $\mathcal{V}$ generates a commitment value $C_{\mathcal{V}} = c_{\mathcal{V}} \cdot G$ where $c_{\mathcal{V}} \in [1, n-1]$ and creates the parameters of the signature pair $(e_{\mathcal{V}}, S_{\mathcal{V}})$. The challenge $e_{\mathcal{V}}$ contains the message $M_{\mathcal{V}}$ concatenated with the $\mathcal{V}$'s random nonce $R_{\mathcal{V}}$, $\mathcal{R}$'s random nonce $N_{\mathcal{R}}$, the $x_{C_{\mathcal{V}}}$ coordinate of the commitment value of $\mathcal{V}$ and the $x_{\overline{C_{\mathcal{R}}}}$ coordinate of the commitment value of $\mathcal{R}$ calculated earlier. $S_{\mathcal{V}}$ forms the response of the signature pair and is given by $S_{\mathcal{V}} = c_{\mathcal{V}} - a \cdot e_{\mathcal{V}} \, (mod\, n)$.

When $\mathcal{R}$ receives the message from $\mathcal{V}$, it first checks the proof $v_{\mathcal{V}}$ for correctness before decrypting and verifying the signature. The proof is validated by taking the hash of the $\mathcal{V}$'s random nonce, $\mathcal{R}$'s random nonce and the $x_{C_{\mathcal{R}}}$ coordinate of

the commitment value of $\mathcal{R}$ generated in message 2. If $\mathcal{V}$ is well-behaved and has verified the signature given by $\mathcal{R}$, the proof $v_{\mathcal{V}}$ should match which implies that $\mathcal{V}$ has committed some CPU resources and is not malicious. If the proof supplied by $\mathcal{V}$ is incorrect, $\mathcal{R}$ will drop the request to establish pairwise keys and terminate the session. If the proof is correct, $\mathcal{R}$ then proceeds to decrypt the received encrypted message using its own private key $k_{\mathcal{R}}^{-}$ to reveal the $LOCM_{\mathcal{V}}$ and the nonce $N_{\mathcal{V}}$. After that, it searches for the public key of $\mathcal{V}$ in the VPFD directory and start to verify the signature pair $(e_{\mathcal{V}}, S_{\mathcal{V}})$ provided by $\mathcal{V}$. The verification begins by computing $\mathcal{V}$'s version of the $\mathcal{V}$'s commitment value $\overline{C_{\mathcal{V}}} = G \cdot S_{\mathcal{V}} + k_1^{+} \cdot e_{\mathcal{V}}$ and then checking if $\overline{e_{\mathcal{V}}} = h(M_{\mathcal{V}}, R_{\mathcal{V}}, N_{\mathcal{R}}, x_{\overline{C_{\mathcal{V}}}}, x_{C_{\mathcal{R}}}) = e_{\mathcal{V}}$ that is received. If the verification of $\mathcal{V}$'s signature is successful, it means that $\mathcal{V}$ is authenticated. In our approach, $\mathcal{R}$ only needs to verify the signature pair $(e_{\mathcal{V}}, S_{\mathcal{V}})$ from $\mathcal{V}$ if the proof $v_{\mathcal{V}}$ is verified true.

With the GPS locations and random nonce values $(LOCM_{\mathcal{V}}, LOCM_{\mathcal{R}}, N_{\mathcal{V}}, N_{\mathcal{R}})$, $\mathcal{V}$ and $\mathcal{R}$ transit into the key derivation module which is described in Section 5.2.4. If any of the verification fails, the session terminates at this stage. Messages 1 to 3 are thus known as the 3 way handshake necessary to mitigate DoS attacks. Once the pairwise keys have been established, $\mathcal{V}$ can use any one of the derived pairwise keys denoted as $K_{\mathcal{V},\mathcal{R}}$ to encrypt the messages via any symmetric algorithm. In addition, HMAC is appended to the message to prove authenticity and integrity. Compared to the digital signature verification, HMAC is more efficient and requires less CPU resources. Timestamp $T_{\mathcal{V}}$ is also included in the message to prevent replay attacks. In order for $\mathcal{R}$ to know which pairwise key is used to decrypt the actual message, the pairwise key is indexed by keyID and sent to $\mathcal{R}$. The format of the message sent by $\mathcal{V}$ to $\mathcal{R}$ is given as:

Vehicle $\mathcal{V} \rightarrow$ RSU $\mathcal{R}$:

$$keyID, Message, T_{\mathcal{V}}, HMAC(K_{\mathcal{V},\mathcal{R}} \parallel Message \parallel T_{\mathcal{V}})_{K_{\mathcal{V},\mathcal{R}}}.$$

When $\mathcal{R}$ receives the message from $\mathcal{V}$, it uses the corresponding pairwise key indexed by the key ID to decrypt the actual message. Only the rightful recipient with the same shared key $K_{\mathcal{V},\mathcal{R}} = K_{\mathcal{R},\mathcal{V}}$ can decrypt the message to retrieve and verify the HMAC value. If the recipient is not able to decrypt the message or if HMAC value is incorrect, $\mathcal{R}$ drops the received packet. Moreover, our scheme can also be applied to the pairwise V2V communications. However, the requesting vehicle has to contact

the RSU first to request for the recipient's public key because the vehicles do not have the VPFD repository. Thereafter, the rest of the procedure in V2V follows the messaging protocol in Figure 5.4.

5.2.3.2 Group Keys Agreement

Our key agreement module can also be extended to support group communications in a V2V context. We illustrate this process in Figure 5.5 where Vehicle $\mathcal{V}_1$ is the group leader trying to establish group keys with Vehicles $\mathcal{V}_2$, $\mathcal{V}_3$, $\mathcal{V}_4$ via the RSU $\mathcal{R}$. The role of $\mathcal{R}$ is to act as a mediator to help authenticate the identities of the requested vehicles on behalf of $\mathcal{V}_1$. Since each RSU has a copy of the VPFD, it can look up the public keys of the vehicles to encrypt messages and verify the digital signatures of the requested vehicles without exchanging certificates.

Message 1: Message 1, message 4 and message 5 are similar to the 3-way handshake discussed in Section 5.2.3.1. They are needed to prevent malicious vehicles from conducting DoS attacks. In message 1, $\mathcal{V}_1$ which is the appointed group leader initiates a request message to $\mathcal{R}$ to establish group keys. This key request message contains $\mathcal{V}_1$'s generated random nonce $R_{\mathcal{V}_1}$ to ensure freshness of the request. It also contains the IDs of all the vehicles that $\mathcal{V}_1$ wants to communicate with in a group.

Message 2 and Message 3: Next, $\mathcal{R}$ retrieves the list of vehicles' ID in message 1 and unicasts a challenge message to all the requested vehicles in the list to prove authenticity. Group ID is added by $\mathcal{R}$ at this stage to identity the communication group. Only vehicles with their corresponding private keys and updated copy of the RSU-PFD can decrypt and verify the messages from $\mathcal{R}$. After that, each requested group vehicles sends a response message back to the $\mathcal{R}$ containing the information depicted in message 3 of Figure 5.5. $\mathcal{R}$ verifies the authenticity of each vehicle by verifying the signature using the corresponding vehicle's public key stored in VPFD.

Message 4: After verification, $\mathcal{R}$ transmits a confirmation message $M_{\mathcal{R}}$ back to $\mathcal{V}_1$. The confirmation message contains a list of vehicles that have passed the authentication process. Vehicles that have failed the authentication cannot take part in the group communication. For example in message 4 of Figure 5.5, $\mathcal{V}_3$ is excluded in

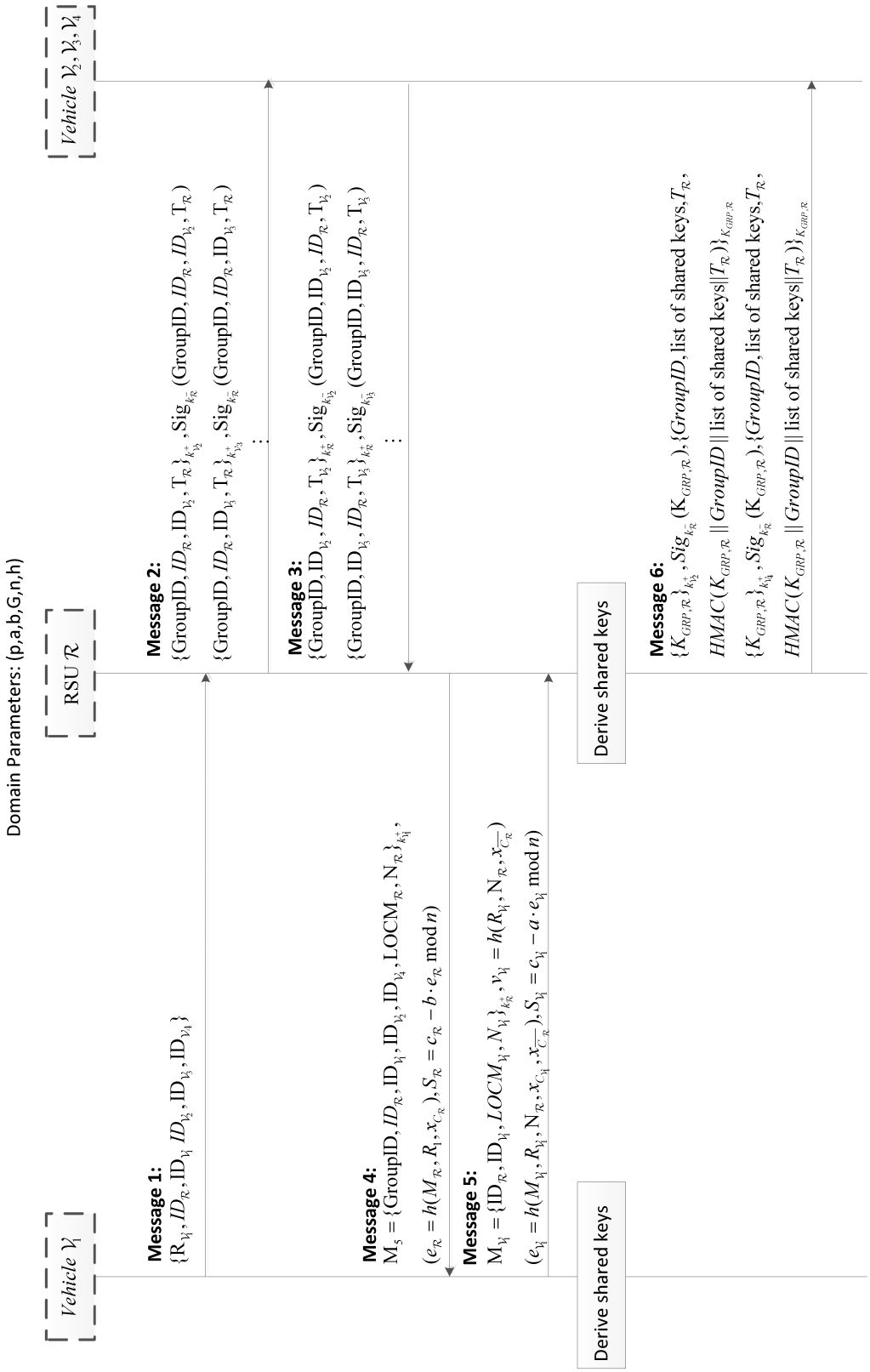

Figure 5.5: Exchange of keying materials for establishing group keys in V2V.

the confirmation message to $\mathcal{V}_1$. In this way, $\mathcal{V}_1$ which is the group leader is better informed of the credibility of the vehicles it wishes to communicate with. $\mathcal{R}$ also provides its GPS location information $LOCM_{\mathcal{R}}$ and a random nonce $N_{\mathcal{R}}$ for deriving the group keys later. The message is encrypted using $\mathcal{V}_1$'s public key. After that, $\mathcal{R}$ creates an EC-Schnorr signature $(e_{\mathcal{R}}, S_{\mathcal{R}})$ for $\mathcal{V}_1$ to verify. When $\mathcal{V}_1$ receives the message from $\mathcal{R}$, it decrypts the message $M_{\mathcal{R}}$ using its own private key to retrieve the list of authenticated vehicles that qualify for group communications, location information $(LOCM_{\mathcal{R}})$ and the random nonce $N_{\mathcal{R}}$ that are used for the key derivation module. Then, it searches for the public key of $\mathcal{R}$ in the RSU-PFD and use the public key together with the received signature pair $(e_{\mathcal{R}}, S_{\mathcal{R}})$ to verify the $\mathcal{R}$'s signature.

Message 5: If the signature verification is successful, $\mathcal{V}_1$ proceeds to compute the proof $v_{\mathcal{V}_1}$. At the same time, $\mathcal{V}_1$ provides its own GPS location given as $LOCM_{\mathcal{V}_1}$ and generates another random nonce $N_{\mathcal{V}_1}$ which is used for deriving the group keys. The entire message $M_{\mathcal{V}_1}$ is encrypted using $\mathcal{R}$'s public key retrieved from the RSU-PFD as discussed before. An EC-Schnorr signature is generated using $\mathcal{V}_1$'s private key and is attached to the reply message back to the $\mathcal{R}$ to guarantee authenticity and message integrity. Before $\mathcal{R}$ proceeds to verify $\mathcal{V}_1$'s signature $(e_{\mathcal{V}_1}, S_{\mathcal{V}_1})$, it first verifies the proof $v_{\mathcal{V}_1}$ provided by the vehicle. If $\mathcal{V}_1$ adheres to the protocol and commit some of its CPU resources to verify the $\mathcal{R}$'s signature in message 4, the proof $v_{\mathcal{V}_1} = h(R_{\mathcal{V}_1}, R_{\mathcal{R}}, x_{\overline{C_{\mathcal{R}}}})$ provided by $\mathcal{V}_1$ should match the proof $\overline{v_{\mathcal{V}_1}} = h(R_{\mathcal{V}_1}, R_{\mathcal{R}}, x_{C_{\mathcal{R}}}$ computed by $\mathcal{R}$. Once the proof is validated in terms of correctness, $\mathcal{R}$ continues to verify $\mathcal{V}_1$'s signature using $\mathcal{V}_1$'s public key available in the VPFD and decrypts the message to retrieve the $LOCM_{\mathcal{V}_1}$ information and random nonce $N_{\mathcal{V}_1}$ for the key derivation module. If the proof is incorrect, $\mathcal{R}$ drops the packet without verifying $\mathcal{V}_1$'s signature. In this way, $\mathcal{R}$ conserves its resources. Once both random nonce values $(N_{\mathcal{V}_1}, N_{\mathcal{R}})$ and $LOCM$ information are known, $\mathcal{V}_1$ and $\mathcal{R}$ transits into key derivation module.

Message 6: After $\mathcal{V}_1$ and $\mathcal{R}$ have derived the group keys, $\mathcal{R}$ helps to distribute the set of common group keys to each authenticated vehicles that $\mathcal{V}_1$ wishes to communicate with. The derived group keys are encapsulated in the message encrypted using a group key $K_{GRP,\mathcal{R}}$ randomly selected from the list of derived group keys.

Each message also contains a HMAC to verify the integrity. In order for each vehicle to decrypt and retrieve the set of group keys, $\mathcal{R}$ encrypts the symmetric key $K_{GRP,\mathcal{R}}$ using the public key of each vehicle and appends a signature to prove its authenticity and integrity. We wish to highlight that only a small amount of information is encrypted using asymmetric encryption. The main bulk of the information is still encrypted using symmetric encryption. Hence, our scheme is much faster and more efficient. Once the list of group keys has been disseminated to the intended recipients in a group, $\mathcal{V}_1$ can use any group key, $K_{GRP,\mathcal{V}_1}$ to encrypt the messages using symmetric algorithm and sent them within a group. HMAC is appended to the message to prove the authenticity and integrity. Only vehicles that have the right group key can decrypt the message. The key index is transmitted along with the message to inform other vehicles in the group which group key to use to decrypt the message. An example of a message from $\mathcal{V}_1$ to the group vehicles is given as:

Vehicle $\mathcal{V}_1 \rightarrow$ Vehicle $\mathcal{V}_2$, $\mathcal{V}_4$:

$$keyID, \{GroupID, Message, T_{\mathcal{V}_1},$$
$$HMAC(K_{GRP,\mathcal{V}_1} \parallel GroupID \parallel Message \parallel T_{\mathcal{V}_1})\}_{K_{GRP,\mathcal{V}_1}}$$

Although this key index is not encrypted, the security of SA-KMP is not compromised because knowing the key index does not reveal the actual group key used for communications.

5.2.4 Key Derivation Module

This module generates the pairwise keys based on an improved 3D grid-key approach. Instead of preloading the keys onto the vehicles and the RSUs, each entity derives the keys dynamically. We illustrate the procedure of obtaining the pairwise keys in a V2I communication where each entity has N plane equations as defined in (5.1).

Both entities have to solve the N plane equations in groups of 3 to determine the solutions. For example, $\mathcal{V}$ inputs its own GPS location $(i_{\mathcal{V}}, j_{\mathcal{V}}, k_{\mathcal{V}})$ into the first plane equation and substitute the GPS location $(i_{\mathcal{R}}, j_{\mathcal{R}}, k_{\mathcal{R}})$ of $\mathcal{R}$ into the second plane equation in (5.1). For the third plane equation, $\mathcal{V}$ uses its own randomly generated nonce value $N_{\mathcal{V}}$ (Message 2 of the key agreement protocol) as the first input for i_i

and the second nonce value $N_{\mathcal{R}}$ generated by $\mathcal{R}$ (Message 3 of the key agreement protocol) as the second input for j_i. The last input k_i of the third equation can be found either by hashing the first and the second nonce values or XOR the two nonce values together.

The random nonce values guarantee the uniqueness of the derived keys. It ensures that another vehicle with the same GPS coordinates communicating with the same RSU will not derive the same keys as the vehicles. With 3 sets of inputs to substitute into the N plane equations, there are $N(N-1)(N-2)$ equation groups to solve which generate $N(N-1)(N-2)$ unique solutions. The set of solutions is described by $S = \{(x,y,z)_1, (x,y,z)_2, \cdots, (x,y,z)_d\}$ where (x,y,z) denotes a solution point and d denotes the number of solutions in the set. To derive the pairwise key at a particular solution point (x,y,z), we define equation (5.4) where the repeated hash results of the concatenation of the two random nonce values and the key requestor's ID is XORed together. In this case, the number of hashing operations is determined by the value of x,y and z in the solution point.

$$\begin{gathered} K_{\mathcal{V},\mathcal{R}}(x,y,z) = H_1^x(N_{\mathcal{V}} \parallel N_{\mathcal{R}} \parallel ID_{\mathcal{V}}) \oplus \\ H_2^y(N_{\mathcal{V}} \parallel N_{\mathcal{R}} \parallel ID_{\mathcal{V}}) \oplus H_3^z(N_{\mathcal{V}} \parallel N_{\mathcal{R}} \parallel ID_{\mathcal{V}}) \end{gathered} \tag{5.4}$$

where $H_\alpha^\beta(\cdot)$ denotes that the hashing function $H_\alpha(\cdot)$ is repeated for β times. Similarly, $\mathcal{R}$ solves $N(N-1)(N-2)$ equation groups and performs the same operations as described in (5.4) to compute the pairwise keys.

It can be shown that for any key, this condition holds: $K_{\mathcal{V},\mathcal{R}} = K_{\mathcal{R},\mathcal{V}}$. Once the pairwise keys are obtained, we can use any key from the $N(N-1)(N-2)$ key pool for the symmetric encryption or decryption. We can also choose a subset of δ keys from the key pool to compute a composite key using the recursive XOR operation as shown in (5.5) where (x,y,z) is an element of the solution set S and contains d solutions.

$$\begin{gathered} K_{composite,\mathcal{V},\mathcal{R}} = \oplus K_{\mathcal{V},\mathcal{R}}(x,y,z) \\ where\,(x,y,z) \in S\,and\,2 \leq \delta \leq d \end{gathered} \tag{5.5}$$

5.2.5 Revocation Module

In our scheme, both the public key request module and the key agreement module rely on the EC-Schnorr signature for mutual authentication. If a vehicle fails the

authentication, the RSU will send a message to the TA to revoke that particular vehicle's public key. TA will update the VPFD and disseminate it to all the RSUs in the region via a secure network in real time. It implies that future communications with that vehicle will be disabled unless it re-registers itself with the TA again to receive another set of public/private key pairs. After that, the new updated VPFD containing the vehicle's new public key will be disseminated to all the RSUs.

On the lifetime of the pairwise keys, we assume that each vehicle and RSU sends a message between 100ms-300ms according to the WAVE standard [4] and [24]. Furthermore, suppose that a shared key can only be used for one message, the lower bound and upper bound of the expiry time of the derived keys can be determined using (5.6).

$$\begin{aligned} exp_{LB} &= (0.1 * \sharp \text{ of common keys}) \text{ in seconds} \\ exp_{UB} &= (0.3 * \sharp \text{ of common keys}) \text{ in seconds} \end{aligned} \tag{5.6}$$

where $\sharp$ of common keys$= N(N-1)(N-2)$ and N is the number of plane equations. If N=10, two communicating nodes get to keep 720 keys for the duration between 72 seconds and 216 seconds before they are destroyed. When the keys expire, the nodes need to re-initiate the key agreement module again.

Using this information, we can further determine how many RSUs need to be informed about the derived keys in a V2I communication. The formula is given in (5.7).

$$\begin{aligned} Dist_{LB} &= (V_{avg} * exp_{LB}) \text{ in meters} \\ Dist_{UB} &= (V_{avg} * exp_{UB}) \text{ in meters} \end{aligned} \tag{5.7}$$

where V_{avg} is the average speed of the vehicle. Suppose the RSUs are 1km apart and the average speed of the vehicle is 100km/h on a highway, the number of RSUs to inform is between 2 and 6. It means that a vehicle does not have to establish shared keys each time it passes an RSU which can greatly reduce the rekeying overhead.

It is also possible to determine the expiry time of a key based on its usage count. We can impose a counter to track the number of messages encrypted using a particular derived key. Once a derived key has exceeded the count limit predefined by the key requestor, another key will be selected from the list. When all the keys are used up or have expired, communicating nodes need to re-initialise the key agreement procedure again.

5.3 Security Analysis

We analyze the SA-KMP scheme to prove that it is secured against classical attacks such as DoS, eavesdropping, data modification, replay, GPS spoofing, node impersonation, repudiation, and collusion attacks. We further provide a formal security validation of our scheme using the ProVerif tool [128].

5.3.1 Classical Attacks

DoS Attack - The purpose of DoS attacks is to make a service or resource unavailable to other legitimate entities. This is achieved by flooding the target destination with many invalid requests in an attempt to overwhelm the system and prevent it from further processing the requests of other legitimate nodes. By the proposed SA-KMP scheme, DoS attacks are mitigated because the RSU has to check if the vehicle has provided the correct hash value which contains the commitment value generated by the RSU in the public key request module. It implies that a vehicle has to spend expensive operations to verify the signature pair provided by the RSU to reveal the correct commitment value. By doing so, it discourages any malicious attackers from launching DoS attacks. On the other hand, the RSU only needs to verify the vehicle's signature when the hash value is evaluated to be true which is inexpensive to the RSU. The malicious vehicle may attempt to skip the computational intensive verification of the RSU's signature by trying to forge a valid hash value containing the commitment value. However, it is not possible due to the property of the one-way hash function.

To further illustrate the effectiveness of our scheme, we calculate the authentication delay when the vehicles flood many signatures to the RSU and compare the results to the PKR scheme [104] and the Wasef's scheme [125]. The authentication delay is defined as the time incurred by the RSU to authenticate the vehicles. In the SA-KMP scheme, the authentication delay is given by the generation and verification times of the EC-Schnorr signature if the hash proof is correct. If the hash proof is incorrect, the RSU will skip the signature verification step and therefore, the authentication delay will consist of only the generation time of the EC-Schnorr signature and the hashing time. In the PKR scheme, the authentication delay is based on the verification of the ECDSA signature while in the Wasef's scheme, the

authentication delay is found by adding up the time taken by ECDSA and HMAC operations. We implement the three schemes in C programming language on a 32 bit Debian Linux operating system with 2 GB RAM running on an Intel Core i7-2620M@2.70Ghz workstation. We ran the simulation for 100000 times to capture the average timings of the different components in these schemes in Table 5.3 and summarize the remaining simulation settings in Table 5.4.

Table 5.3: Authentication delays of the different components for different schemes

Algorithm	**SA-KMP**	**PKR**	**Wasef's Scheme**
Signature Generation (ms)	0.154	-	-
Signature Verification (ms)	1.888	2.834	2.834
Generate/Verify Proof (ms)	0.155	-	-
HMAC (ms)	-	-	0.008

Table 5.4: Simulation settings

Parameters	**Value**
Message Load	60 to 200
Invalid messages	10% and 30% of the message load

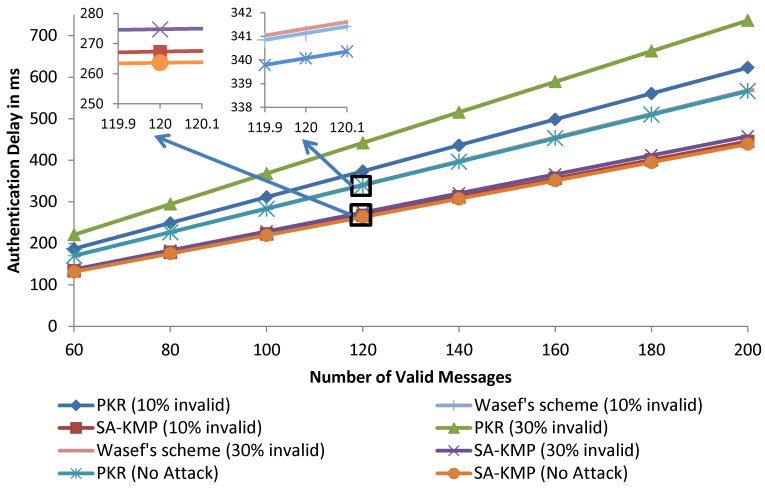

Figure 5.6: Authentication delay under different scenarios.

Figure 5.6 shows the authentication delay as a function of message load for two cases: 1) no DoS and 2) DoS attacks. The DoS attack is simulated by injecting 10% and 30% of the message load as invalid messages. We observe that when the PKR scheme is deployed, the introduction of invalid messages (10% and 30% of the number of valid messages) increases the authentication delay substantially because the PKR scheme has no defense mechanism to mitigate the DoS attacks. Consequently, the PKR scheme has to verify all the signatures even if they are invalid. On the other hand, the Wasef's scheme appends an HMAC to all the outgoing messages when the number of invalid signatures to the number of valid signatures exceeds a threshold. Therefore, the authentication delay is slightly higher than the PKR scheme where it consists of the sum of all the ECDSA verification times and the HMAC verification times. In contrast, the authentication delay for the SA-KMP scheme is significantly lower than that of the PKR scheme and Wasef's scheme under no DoS attack. Furthermore, according to the left inset figure in Figure 5.6, the performance of the SA-KMP scheme when subject to 10% and 30% invalid messages, has little effect on the authentication delay and is also significantly lower than the PKR scheme and the Wasef's scheme under DoS attacks. The SA-KMP scheme is efficient because the RSU only has to perform a light step to validate the hash proof containing its commitment value. If the proof is incorrect, RSU will skip the signature verification. The authentication delay of our scheme, in this case, consists of the sum of the signature generation time and the hashing time. A lower authentication delay indicates a higher availability to service the requests of other vehicles. Based on these observations, we conclude that SA-KMP does not introduce additional delay and can mitigate DoS attackers effectively even though security measures are introduced.

Eavesdropping Attack - This is a type of passive attacks where the eavesdroppers monitor the communication channel to access sensitive or personal information without authorization. This form of attack is harder to detect because it does not involve any alteration of data. SA-KMP is secure against eavesdropping attack because all the messages are encrypted to ensure confidentiality. In our scheme, if m is selected to be a prime and the rank of the coefficient matrix is of full rank, then two nodes have in common $N(N-1)(N-2)$ distinct keys where N is the number of plane equations. This implies that the chances of guessing the correct derived key

is $\frac{1}{N(N-1)(N-2)}$ which is very low if N is large. Moreover, if the SHA-1 encryption is used, each derived key is 160 bits in length. The computational complexity to find the correct key is therefore $O(2^L)$ where L = the length of the key in bits. Furthermore, two communicating entities can derive a composite session key using some of the derived keys to increase the difficulty of cracking the keys. For example, if N=10, then there are 720 derived keys between two communicating entities. If any 50 keys are selected to compose the composite key, there will be at least $\binom{750}{50}$ possible combinations of the composite keys.

Data Modification Attack - This is a type of active attack in which the information is modified by the attackers. A malicious vehicle can delete, alter or forged information without any authorization before forwarding it to the next recipient. For example, a vehicle sends a warning message to another vehicle to warn that there is congestion ahead. The malicious vehicle intercepts this message and modifies it to indicate there is no congestion. By doing this, the victim vehicle is then caught in a traffic jam which leads to longer delay or even accidents. By the proposed SA-KMP scheme, all the messages in the public key request module and the key agreement module are digitally signed to detect data modifications. Furthermore, in the key agreement module, when the shared keys have been established between two entities or among a group, a HMAC using one of the derived keys is appended to the message to check for the integrity and authenticity of the message.

Replay Attack - In this attack, the malicious vehicle saves a copy of a message that is already sent and retransmits at a later stage. To solve this problem, random nonce values are used in the public key request module including the key agreement module. The nonce is at least 128 bits long which implies that the probability of getting the same random nonce for two different communication sessions is equal to $\frac{1}{2^{128}}$. Also, all the messages are time-stamped to ensure the freshness. If the message is not received within a tolerable period, HMAC verification will fail, and the packet will be discarded.

GPS Spoofing Attack - In this attack, the malicious vehicle may try to subvert the key agreement scheme by providing false location information to the other com-

munication node when establishing the shared keys. This type of attack can be performed by external attackers or internal attackers. In the SA-KMP, the external attackers are not able to intercept the message and modify the GPS coordinates because all the messages are encrypted using asymmetric cryptography during the key agreement phase. Furthermore, the private key is protected and stored in the HSM which is programmed to self-erase when any physical tampering is detected. On the other hand, an internal attacker may misbehave and modify the GPS coordinates in the $LOCM$ message. This form of attack can be prevented because the vehicles and the RSUs are equipped with a GPS receiver to perform cross verification on the received GPS coordinates. It can also be avoided by employing the verifiable multilateration method [129] to verify the vehicles' positions. Hence, a GPS spoofing attack cannot succeed.

Node Impersonation Attack - This attack is also known as Sybil attack and can be carried out in several ways depending on the intent of the attacker. A malicious vehicle can fake a legitimate ID to deceive other vehicles or RSUs that he/she is the real originator of the message. The malicious vehicle can also forge IDs to evade detection upon accidents or crimes. By the proposed SA-KMP scheme, the vehicles, and the RSUs are given a public key repository where it contains the certified node ID and public key bindings issued by the TA during the registration phase. This information is used to authenticate each other mutually in the key agreement step. Therefore, impersonation cannot succeed. The position of the nodes is also verified in the process which makes the impersonation even harder. Using shared keys for the symmetric encryption can also provide some level of authentication as the shared keys are only known to a group of vehicles. Therefore, each party can be sure that it is communicating to the correct party and not some impersonators.

Repudiation Attack - In this attack, the attacker denies having performed an action such as sending or receiving a message. By the proposed SA-KMP scheme, non-repudiation is achieved by employing digital signature in the public key request module and the key agreement module as well as requesting the vehicle to sign the key index of the shared key.

Collusion Attack - By this attack, two or more vehicles may collude to capture the nodes to compromise the session keys or on a larger scale, to capture the whole key space. For this reason, the grid-based key [37, 38] and the probabilistic key sharing [88] schemes are required to change the keys periodically which increase the communication overhead. In the design of the proposed SA-KMP scheme, the keys are not preloaded to increase the resilience to node capture attacks. Instead, the key derivation module generates the keys on demand based on the current inputs to the plane equations. Even if a particular group of network entities is compromised, it only affects the communication in that particular region without jeopardizing the whole network. Moreover, the public key request module of the proposed SA-KMP scheme is resilient against the collusion attacks. More specifically, each entry ID,public key in the public key directory is attached with a signature generated by the TA's signing key during registration phase and the RSUs are not given the TA's signing key. Therefore, there is no way for the RSU to collude with another vehicle to manipulate and issue a valid signed public key reply. Another advantage of storing the TA's signature is that RSU do not need to generate a signature when it issues the requested public key. It can simply retrieve the TA's signature from the stored entry and send it back to the requesting vehicle thereby reducing the workload on the RSU.

5.3.2 Formal Verification by ProVerif

ProVerif [128] is a software tool designed to analyze the security properties of a cryptographic protocol automatically. It can support the modeling of several cryptographic primitives, including symmetric and asymmetric encryption, digital signatures, hash functions, bit-commitment, and non-interactive zero-knowledge proofs. It is capable of proving reachability properties, correspondence assertions, and observational equivalence which allows for the analysis of secrecy, authentication and privacy properties. Besides that, other properties such as traceability and verifiability are supported.

In protocol analysis, ProVerif considers an attacker in the Dolev-Yao model [130] where the attacker has complete control over the communication channel and can read, modify, delete, and inject messages. In addition, ProVerif can capture the behavior of dishonest participants in cryptographic protocols. During protocol analysis, an unbounded number of sessions and an unbounded message space are analyzed

to prove the desired properties. When a property cannot be proven, ProVerif supports an attack reconstruction in which it attempts to reconstruct the attack trace that falsify the desired property. By doing this, ProVerif provides designers with a way to identify security flaws in their protocol design and enables them to rectify the vulnerabilities. In this section, we use ProVerif to validate the reachability and correspondence assertions of the key agreement module presented in Figure 5.4.

5.3.2.1 Modeling of Key Agreement Module

The modeling of the key agreement module using ProVerif can be divided into four parts namely, the declarations, test statements, process macros and the main process. With reference to the ProVerif code provided in Listing 5.1, we declare six types of variables from Lines 1 to 6. We define EC point, host identifiers, nonce, GPS location and derived key to differentiate the different inputs in our program. We further declare the identifiers of vehicle $\mathcal{V}$ and RSU $\mathcal{R}$ namely id_v and id_rsu as constants in line 7 and line 8. Next, we declare the free channel c in line 9 which is used by $\mathcal{V}$ and $\mathcal{R}$ to communicate.

```
1  type point.
2  type v_host.
3  type rsu_host.
4  type nonce.
5  type gps_location.
6  type key.
7  const id_v:v_host.
8  const id_rsu:rsu_host.
9  free c:channel.
10
11 fun hash(bitstring):bitstring.
12 fun scalar_point_mult(bitstring,point):point.
13 fun x_coordinate(point):bitstring.
14 fun mult(bitstring,bitstring):bitstring.
15 fun schnorr_S_gen(bitstring,bitstring):bitstring.
16 fun sum_point(point,point):point.
17 fun encr(bitstring,point):bitstring.
18 reduc forall P:point,s:bitstring,m:bitstring; decr(encr(m,scalar_point_mult(s,P)),s)=m.
19 fun senc(bitstring,key):bitstring.
20 reduc forall m:bitstring,k:key; sdec(senc(m,k),k)=m.
21 fun repeat_hash(bitstring):bitstring.
22 fun xor(bitstring,bitstring):bitstring.
23 equation forall x:bitstring,y:bitstring;xor(xor(x,y),y)=x.
```

```
24 equation forall x:bitstring; xor(x,xor(x,x))=x.
25 equation forall x:bitstring; xor(xor(x,x),x)=x.
26 equation forall x:bitstring,y:bitstring;xor(y,xor(x,x))=y.
27 fun bitstring_to_key(bitstring):key [data,typeConverter].
28 equation forall s:bitstring, m:bitstring, x:bitstring, P:point;
29 sum_point(scalar_point_mult(schnorr_S_gen(x, mult(s, hash((m, x_coordinate(scalar_point_mult(x,P))
       )))), P),
30 scalar_point_mult(hash((m,x_coordinate(scalar_point_mult(x,P)))),scalar_point_mult(s,P)))
31 = scalar_point_mult(x,P).
32 table RSU_PFD_keys(rsu_host,point).
33 table VPFD_keys(v_host,point).
```

Listing 5.1: Declaration part of keyagreement.pv

After that, we specify a list of cryptographic functions denoted by the keyword *fun* to model the symmetric encryption, asymmetric encryption including the generation and verification of the EC-Schnorr digital signature. For example, Line 11 defines the hash function as a unary constructor hash. Lines 12-16 define the operations to generate an EC-Schnorr signature. Asymmetric encryption/decryption functions including symmetric operations are defined from Line 17 to Line 20. Lines 21-22 denote the operations to compute the shared keys. More formally, we declare repeat_hash function in Line 21 to represent the repeated hashing of the two random nonce values concatenated with the requested ID. Line 22 together with the following lines of code 23-26 defines the truth table of the XOR operation and is used for computing the derived shared key. Line 27 is a type converter that takes inputs of type bitstring and returns a result of type key. This is a feature of ProVerif to ensure that there will not be any type mismatch when applying functions to different arguments. Equation 28-31 defines the reduction rule of the EC-Schnorr signature verification operation. Lastly, Lines 32 and 33 declare two tables namely, RSU-PFD and VPFD respectively that relate the host IDs with their corresponding public keys. In our implementation, vehicles and RSUs who are taking part in the key agreement protocol have to look up the public key of the corresponding host in their respective tables in order to verify or encrypt messages.

The second part of the program as depicted in Listing 5.2 defines a list of statements to prove the security properties of the key agreement module.

```
1  event vSends1(nonce).
2  event rsuSends2(nonce,nonce).
3  event vSends3(nonce,nonce,bitstring).
4  event termRSU(nonce,bitstring,key).
5  event termVehicle(key).
6
7  query rand:nonce,rand1:nonce,rand2:nonce,rand3:bitstring,k1:key;
8  inj-event(termVehicle(k1)) ==> (inj-event(termRSU(rand2,rand3,k1)) ==> (inj-event(vSends3(rand1,
       rand2,rand3)) ==> (inj-event(rsuSends2(rand,rand1)) && inj-event(vSends1(rand))))).
9  query rand:nonce,rand1:nonce,rand2:nonce,rand3:bitstring,k1:key;
10 inj-event(termRSU(rand2,rand3,k1)) ==> (inj-event(vSends3(rand1,rand2,rand3)) ==> (inj-event(
       rsuSends2(rand,rand1)) && inj-event(vSends1(rand)))).
11
12 free secretmessage:bitstring [private].
13 query attacker(secretmessage).
```

Listing 5.2: Query part of keyagreement.pv

More specifically, we define five events (Lines 1 to 5) which we also indicate in red in Figure 5.4, as correspondence assertions to prove the proper sequencing of events during the protocol execution. That is, if an event has been executed, it implies that another event has been previously executed. Such an ordered association between events is a standard way of specifying authentication. These events are described as follows:

- event vSends1($R_{\mathcal{V}}$) - which denotes that vehicle $\mathcal{V}$ has initiated a request with RSU $\mathcal{R}$ with the supplied nonce value $R_{\mathcal{V}}$.
- event rsuSends2($R_{\mathcal{V}}$,$N_{\mathcal{R}}$) - is used to denote that RSU $\mathcal{R}$ has accepted to run the protocol with vehicle $\mathcal{V}$ with the proposed $R_{\mathcal{V}}$ as the first argument and $\mathcal{R}$'s nonce value $N_{\mathcal{R}}$ as the second.
- event vSends3($N_{\mathcal{R}}$,$N_{\mathcal{V}}$,$v_{\mathcal{V}}$) - means that vehicle $\mathcal{V}$ has also accepted to run the protocol with RSU $\mathcal{R}$ with the supplied hash proof $v_{\mathcal{V}}$ and its own nonce $N_{\mathcal{V}}$.
- event termRSU($N_{\mathcal{V}}$,$v_{\mathcal{V}}$,$K_{\mathcal{R}}$) - denotes the RSU $\mathcal{R}$'s belief that he has terminated a protocol run with vehicle $\mathcal{V}$ successfully by accepting the hash proof $v_{\mathcal{V}}$ and successfully derived the shared key $K_{\mathcal{R}}$ using $N_{\mathcal{V}}$.
- event termVehicle($K_{\mathcal{V}}$) - denotes that vehicle $\mathcal{V}$ has terminated a protocol run with RSU $\mathcal{R}$ with the derived shared key as the first argument.

We note that the arguments within the parentheses of each event as shown above are specified according to the exact variables presented in Figure 5.4 for clarity purposes. In our implementation using ProVerif tool, these arguments have been replaced appropriately by their user defined types i.e. nonce and bitstring in the program for simplicity sake. After the definition of events, we formalize a query statement (Lines 7 to 8) to test the authenticity of RSU $\mathcal{R}$ as follows:

query rand:nonce, rand1:nonce, rand2:nonce, rand3:bitstring, k1:key;
inj-event(termVehicle(k1))==>(inj-event(termRSU (rand2,rand3,k1))
==>(inj-event(vSends3(rand1,rand2,rand3))==>
(inj-event(rsuSends2(rand,rand1)) && **inj-event (vSends1(rand)))))**.

The above relationship implies that each termination of the vehicle has to be preceded by an instance of the following events: termRSU(), vSends3(), rsuSends2() and vSends1(). If the query result is true, it means that the RSU is properly authenticated. Otherwise, it is not. Similarly, we formalize a second query statement as defined in Lines 9 and 10 of Listing 5.2 to test the authenticity of the vehicle $\mathcal{V}$. Based on the property of correspondence assertions, it can be proved that vehicle $\mathcal{V}$ is authenticated only if RSU $\mathcal{R}$ has terminated the protocol followed by the successful execution of events vSends3(), rsuSends2(), and vSends1().

query rand:nonce, rand1:nonce, rand2:nonce, rand3:bitstring, k1:key;
inj-event(termRSU(rand2,rand3,k1)) ==>
(inj-event(vSends3(rand1,rand2,rand3))
==>**(inj-event (rsuSends2 (rand,rand1))** && **inj-event(vSends1(rand))))**.

For both correspondence queries above, the injective correspondence is used to capture the one-to-one relationship between the number of protocol runs performed by each participant. When the injective query is violated, it means that the protocol is subject to a replay attack and besides failing the authenticity property. Furthermore, we define a global free name *secretmessage* of type bitstring in Line 12 for testing the secrecy of the derived key. Subsequently, we test the secrecy of the derived key by defining a query statement (Line 13) as follows:

query attacker(secretmessage).

The third part of the program contains the sub-process macro of vehicle $\mathcal{V}$ and RSU $\mathcal{R}$. It describes the message exchanges between $\mathcal{V}$ and $\mathcal{R}$ in a V2I communication with reference to Figure 5.4.

```
1  let rsu(priv_rsu:bitstring,G:point)=
2  in(c,hostY:v_host);
3  if hostY = id_v then
4  get VPFD_keys(=hostY,Pub_v) in
5  in(c,(r1:nonce,id_x:v_host,id_y:rsu_host));
6  if id_x=hostY && id_y=id_rsu then
7  new N_rsu:nonce;
8  new c_rsu:bitstring;
9  new LOCM_rsu:gps_location;
10 event rsuSends2(r1,N_rsu);
11
12 out(c,(encr((id_x,id_y,LOCM_rsu,N_rsu),Pub_v), hash((encr((id_x,id_y,LOCM_rsu,N_rsu),Pub_v),r1,
       x_coordinate(scalar_point_mult(c_rsu,G)))), schnorr_S_gen(c_rsu,mult(priv_rsu, hash((encr((
       id_x,id_y,LOCM_rsu,N_rsu),Pub_v),r1,x_coordinate(scalar_point_mult(c_rsu,G))))))));
13 in(c,(x:bitstring,proof_v:bitstring,y:bitstring,z:bitstring));
14 let (=hash((r1,N_rsu,x_coordinate(scalar_point_mult(c_rsu,G)))))=proof_v in
15 let x_c_v=x_coordinate(sum_point(scalar_point_mult(z,G),scalar_point_mult(y,Pub_v))) in
16 let (=hash((x,r1,N_rsu,x_c_v,x_coordinate(scalar_point_mult(c_rsu,G)))))=y in
17 let (=hostY,=id_rsu,LOCM_v:gps_location,N_v:nonce)=decr(x,priv_rsu) in
18 let key_rsu_1=repeat_hash((N_v,N_rsu,hostY)) in
19 let key_rsu_2=repeat_hash((N_v,N_rsu,hostY)) in
20 let key_rsu_3=repeat_hash((N_v,N_rsu,hostY)) in
21 let derived_key_rsu=bitstring_to_key(xor(xor(key_rsu_1,key_rsu_2),key_rsu_3)) in
22 event termRSU(N_v,proof_v,derived_key_rsu);
23 out(c,senc(secretmessage,derived_key_rsu)).
```

Listing 5.3: Sub-process macros of RSU $\mathcal{R}$ in keyagreement.pv

Listing 5.3 describes the sub-process macro of RSU $\mathcal{R}$. Lines 2 to 4 denotes the receipt of vehicle identifier id_v and the retrieval of id_v's public key from the VPFD. The retrieval of public key is necessary for the verification of id_v's signature later. Lines 5 to 10 denotes the receipt of id_v's request to negotiate pairwise keys which includes the generation of nonce, commitment values and GPS location by RSU $\mathcal{R}$ to prepare for the key establishment phase. Lines 12 to 17 corresponds to Message 2 in Figure 5.4 where it consists of generating an EC-Schnorr signature for vehicle id_v to verify. These lines of code also include the verification of the hash proof

supplied by id_v. If the hash proof is correct, RSU $\mathcal{R}$ then proceeds to verify the signature of id_v and if the proof is valid, RSU $\mathcal{R}$ decrypts the message using his own public key to retrieve the GPS location and the random nonce of id_v. Lines 18 to 22 of the code describes the operation of deriving the pairwise keys. More specifically, Lines 18,19 and 20 represent the repeated hashing of the random nonce values and the identities based on the solution points to the system of plane equations. Line 21 denotes the derivation of the pairwise keys by XOR-ing the repeated hash results along the x, y, z axis together. Line 23 outputs a message *secretmessage* encrypted using a derived key by the RSU onto the channel c. Two events that we have defined in the declaration part of the program are also introduced in Line 10 and Line 22 of the sub-process program to prove correspondence assertions.

The sub-process of vehicle $\mathcal{V}$ is presented in Listing 5.4. Similar to RSU $\mathcal{R}$, Lines 2 to 4 denotes the retrieval of RSU $\mathcal{R}$'s public key. Lines 5 to 7 refers to the setting up of request message to negotiate pairwise keys with RSU $\mathcal{R}$. Lines 9 to 10 denotes the verification of RSU $\mathcal{R}$'s signature. Line 11 represents the decryption of the message supplied by RSU $\mathcal{R}$ to retrieve the GPS location and random nonce of RSU $\mathcal{R}$. Line 16 denotes the generation of the hash proof using the RSU's commitment value and the random nonce values of vehicle $\mathcal{V}$ and RSU $\mathcal{R}$. Line 18 corresponds to Message 3 in Figure 5.4 which contains its own signature and the generated hash proof. Lines 20 to 23 denotes the pairwise keys derivation using repeated hashing functions and XOR operation as per described earlier. Line 24 checks the correctness of the derived key by decrypting the message sent by RSU $\mathcal{R}$. Furthermore, three other events are defined in the sub-process to capture the relationships of events between vehicle $\mathcal{V}$ and $\mathcal{R}$ to test the authenticity properties. Additionally, Lines 28 to 34 describes the key registration process of the vehicle $\mathcal{V}$ and $\mathcal{R}$.

```
1  let v(G:point,priv_v:bitstring)=
2  in(c,(hostX:rsu_host));
3  if hostX=id_rsu then
4  get RSU_PFD_keys(=hostX,Pub_rsu) in
5  new r1:nonce;
6  event vSends1(r1);
7  out(c,(r1,id_v,hostX));
8  in(c,(x:bitstring,y:bitstring,z:bitstring));
9  let x_c_rsu=x_coordinate(sum_point(scalar_point_mult(z,G),scalar_point_mult(y,Pub_rsu))) in
```

```
10 let (=hash((x,r1,x_c_rsu)))=y in
11 let (=id_v,=hostX,LOCM_rsu:gps_location,N_rsu:nonce)=decr(x,priv_v) in
12 new LOCM_v:gps_location;
13 new N_v:nonce;
14 new c_v:bitstring;
15 new proof_v:bitstring;
16 let proof_v = hash((r1,N_rsu,x_c_rsu)) in
17 event vSends3(N_rsu,N_v,proof_v);
18 out(c,(encr((id_v,hostX,LOCM_v,N_v),Pub_rsu),proof_v,hash((encr((id_v,hostX,LOCM_v,N_v),Pub_rsu),
     r1,N_rsu,x_coordinate(scalar_point_mult(c_v,G)),x_c_rsu)),schnorr_S_gen(c_v,mult(priv_v, hash
     ((encr((id_v,hostX,LOCM_v,N_v),Pub_rsu),r1,N_rsu,x_coordinate(scalar_point_mult(c_v,G)),
     x_c_rsu))))));
19 in(c,(x1:bitstring));
20 let key_v_1=repeat_hash((N_v,N_rsu,id_v)) in
21 let key_v_2=repeat_hash((N_v,N_rsu,id_v)) in
22 let key_v_3=repeat_hash((N_v,N_rsu,id_v)) in
23 let derived_key_v=bitstring_to_key(xor(xor(key_v_1,key_v_2),key_v_3)) in
24 let y1=sdec(x1,derived_key_v) in
25 event termVehicle(derived_key_v);
26 out(c,senc(secretrsu,derived_key_v)).
27
28 let RSU_Registration=
29 in(c,(h:rsu_host,k:point));
30 if h <> id_rsu then insert RSU_PFD_keys(h,k).
31
32 let Vehicle_Registation=
33 in(c,(h:v_host,k:point));
34 if h <> id_v then insert VPFD_keys(h,k).
```

Listing 5.4: Sub-process macros of vehicle $\mathcal{V}$ in keyagreement.pv

---

The last part of the program describes the main process. As shown in Listing 5.5, the main process begins by constructing the public/private key pair for the RSU and vehicle. After that, each entity registers its own public key with the TA where the TA stores the keys into their respective tables. This operation is initiated by the following two commands *insert VPFD_keys(id_v,Pub_v)* and *insert RSU_PFD_keys(id_rsu,Pub_rsu)* in Line 5 and 6 respectively. Following that, an unbounded instances of sub-processes rsu, v, RSU_Registration and Vehicle_Registration are instantiated in parallel to kick start the verification process.

```
1 process
2 new Q:point;
3 new s_rsu:bitstring;
```

```
4  new s_v:bitstring;
5  let P_rsu=scalar_point_mult(s_rsu,Q) in insert RSU_PFD_keys(id_rsu,P_rsu);
6  let P_v=scalar_point_mult(s_v,Q) in insert VPFD_keys(id_v,P_v);
7  (!rsu(s_rsu,Q))|(!v(Q,s_v))|(!RSU_Registration)|(!Vehicle_Registation)
```

Listing 5.5: Main process in keyagreement.pv

5.3.2.2 Verification Results

The execution of the program (***keyagreement.pv***) using the command proverif keyagreement.pv | grep 'RES 'produces the results in the Figure 5.7.

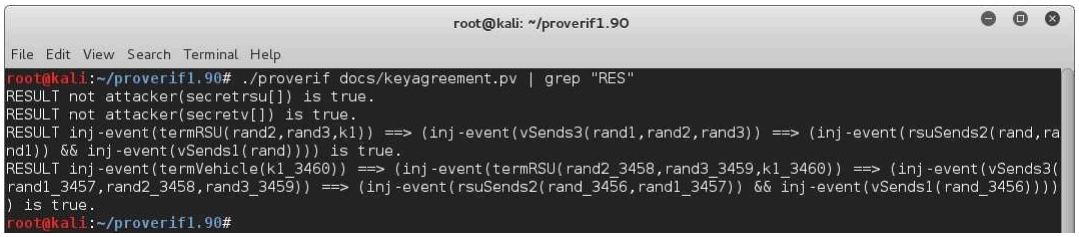

Figure 5.7: Proverif verification results.

- Lines 1-2 indicate that the attacker is unable to determine the derived key shared between the RSU and the vehicle. Hence, the secrecy of the derived key is preserved.
- Lines 3-7 indicate that the injective authentication of the vehicle to the RSU and vice versa hold which means that the RSU and the vehicle participating in the protocol are both authenticated. In addition, it proves that the replay and the impersonation attacks are impossible to succeed by the attacker. As such, we conclude that our key agreement protocol is secure against an active attacker under the Dolev-Yao model.

5.4 Performance Analysis

In this section, we evaluate the performance of the SA-KMP scheme in terms of the transmission and storage overhead, latency, scalability, key generation time and computational complexity.

5.4.1 Transmission Overhead

We analyze the transmission overhead of the SA-KMP scheme and compare its performance with the certificate-based scheme in [29] and the PKR scheme in [104]. The comparison is made by evaluating the transmission overhead when a vehicle sends a packet to an RSU or vice versa.

By the certificate-based scheme, the size of the certificate for a vehicle is 126 bytes and the size of the ECDSA signature that is created by a vehicle using a key length of 224 bits is 56 bytes. The total transmission overhead of a message is, therefore, equal to 126+56 bytes=182 bytes as shown in Table 5.5. By the SA-KMP scheme and the PKR scheme, the vehicles and the RSUs are given a read-only copy of the directory which contains public keys that have previously been verified by the TA. Therefore, there is no need to attach the certificate in the message transmission which means that the total transmission overhead is only 56 bytes as given in Table 5.5.

Table 5.5: Comparison of transmission overhead for various schemes

Scheme	**Sending a message**
Certificate-based PKI [29]	182 bytes
SA-KMP	56 bytes
PKR [104]	56 bytes

We further illustrate the reduction of the transmission overheads in Figure 5.8 by plotting the transmission overhead received by an RSU as a function of increasing the message load from 0 to 30000. It is observed that both the SA-KMP and the PKR schemes have the lowest transmission overhead compared to that of the certificate-based scheme even when the number of messages received at the RSU increases. The transmission overhead of the SA-KMP is 30.8% that of the certificate-based scheme which means that our scheme saves about 69.2% of the communication bandwidth without transmitting certificates.

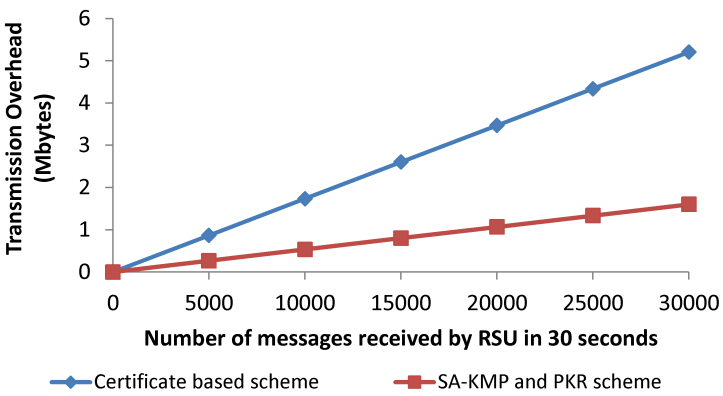

Figure 5.8: Transmission overhead as a function of messages.

5.4.2 Storage Overhead

We estimate the storage required by each RSU and vehicle to store the VPFD and RSU-PFD repositories. According to [29], the public key size of an RSU and a vehicle is 33 bytes and 29 bytes respectively, and the size of the TA's ECDSA signature is 64 bytes. If 4 bytes are used to store the vehicle ID and assuming there are 1 million vehicles in the network, the size of the VPFD held by each RSU would be 1 million x (29+4+64)bytes=97 Mbytes as shown in Table 5.6, which is higher than 33 Mbytes required by the PKR scheme. The increase in storage requirements is due to storing the TA's signature into the database which is used to prevent the collusion attacks.

Table 5.6: Comparison of storage overhead for various schemes

Repository / Scheme	RSU-PFD	VPFD
SA-KMP	3.7 Mbytes	97 Mbytes
PKR [104]	-	33 Mbytes
Certificate-based PKI [29]	126 bytes per certificate x no. of certificates	

On the other hand, if there are 100,000 RSUs deployed in the network, the size of the RSU-PFD which is maintained by each vehicle is equal to 100,000 x (33+4)bytes=3.7 Mbytes as shown in Table 5.6. Assuming the storage capacity of each vehicle and each RSU is 256M bytes, the storage requirements of 97M bytes and

3.7M bytes take up only about 37.9% and 1.45% of the storage space in each RSU and vehicle, respectively which is reasonable. This additional storage requirement is a trade-off for the enhanced security against the collusion attacks and more efficiency in terms of lower transmission overhead. In contrast, the certificate-based PKI scheme requires each vehicle to store a set of certificates for the PKI support as suggested by [31] and [131]. The number of certificates to store, however, is not specified and can vary according to various application requirements. Therefore, the storage overhead is expressed as a function of the number of certificates. Even if the storage requirement is lower than that of the our scheme, we note that the certificate-based PKI scheme has the disadvantage of increased latency due to the need to validate the certificates.

5.4.3 Latency Analysis

Next, we analyze the latency of our scheme and compare the results to the certificate-based PKI scheme [29] and the PKR scheme [104]. We define the latency as the amount of time a packet takes to travel from the source to the destination including the time taken to acquire the public key for verifying the message. In our analysis, we omit the transmission delay and the queuing delay and evaluate the latency as the sum of the **propagation delay** and the **processing delay**. Figure 5.9 illustrates the timing diagrams of the various schemes whereby the vertical arrows represent the processing delay while the diagonal or horizontal arrows represent the propagation delay.

According to Figure 5.9(i), a vehicle has to send a message to an RSU to request for the public key of the target vehicle. To mitigate any DoS attacks during the request of public key, both vehicle and RSU exchange EC-Schnorr signatures which involve verification of proof. Thus, the main bulk of the processing delay in a V2V communication consists mainly of signature generation and verification times including the generation and verification of proof from both sides. On the other hand, the processing delay in a V2I communication is much simpler as the vehicle and RSU has a copy of the RSU-PFD and VPFD respectively which eliminates the need to request a public key. Hence, the processing delay is $T_{search} + T_{verify,ECDSA} + T_{sign,ECDSA}$ while the propagation delay is $T_{tx,signed\,message\,without\,cert.}$. By the certificate-based PKI as shown in Figure 5.9(ii), the main delay is due to the dissemination of CRL

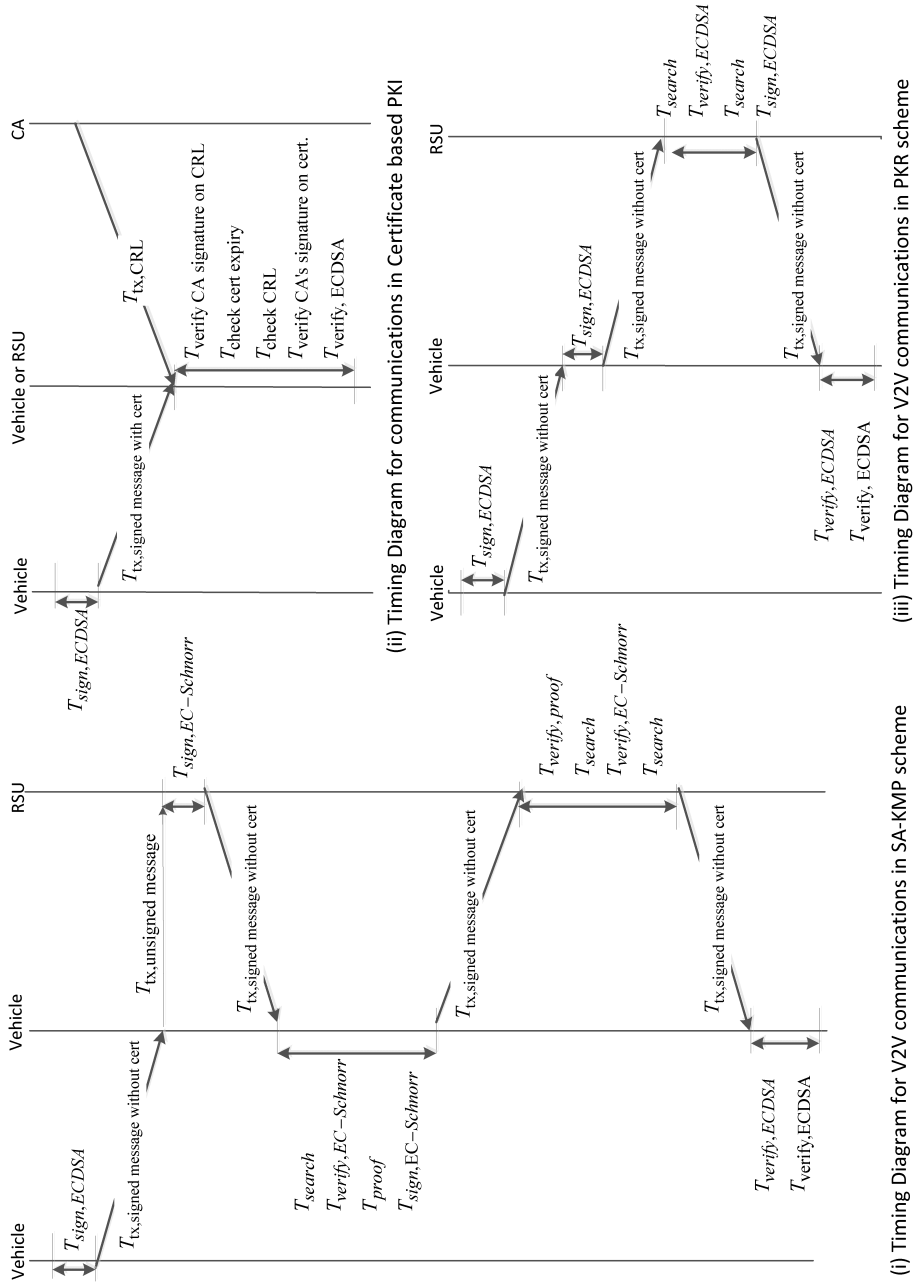

Figure 5.9: Latency of SA-KMP, certificate-based PKI and PKR schemes.

and the verification of the certificates. Lastly, in the PKR scheme as illustrated in Figure 5.9(iii), the latency consists of only the delay in acquiring the public key from the RSU which involves generation and verification of an ECDSA signature.

Calculation of Processing Delay

To assess the latency of each scheme numerically, we first evaluate the **processing delay** by measuring the signature generation and verification times of the EC-Schnorr and ECDSA signature schemes. The EC-Schnorr signature scheme is used by the SA-KMP scheme, while the ECDSA scheme is used by the PKR and the certificate-based PKI schemes. We implement the EC-Schnorr and the ECDSA signature schemes in C language using the Openssl package and ran a total of 100000 simulation on a 32 bit Debian Linux operating system with 2 GB RAM running on an Intel Core i7-2620M@2.70Ghz workstation to compute the average timing. The processing delay also includes the time needed to search for a public key (T_{search}) in the public file directory of the SA-KMP and PKR schemes as well as the time taken to check the validity of certificate against the CRL ($T_{checkCRL}$) in the certificate-based PKI approach. Table 5.7 summarizes all the timings necessary for computing the processing delay.

Table 5.7: Average timings for calculating processing delay

Algorithm / **Operations**	**EC-Schnorr SA-KMP**	**ECDSA PKR**	**ECDSA PKI**
Signing (ms)	0.154	1.486	1.486
Verification (ms)	1.888	2.834	2.834
Generate/Verify Proof (ms)	0.155	-	-
T_{search} (ms)	0.000859	0.000859	-
$T_{checkCRL}$ (ms)	-	-	97.2975

Calculation of Propagation Delay

Next, we consider the following message size given in Table 5.8 to evaluate the **propagation delay** of the three schemes. We treat the size of an unsigned message as 69 bytes and the size of an EC-Schnorr or ECDSA signature as 56 bytes. Thus, the size of a signed message will be 69 bytes + the size of ECDSA signature which

is 125 bytes. If a certificate is attached to the signed message, where the size of a certificate is approximately 126 bytes, the size of a message containing the certificate will be 251 bytes in total. We also consider the size of the CRL which is 0.9 Mbytes to account for the propagation delay in the certificate-based PKI approach. Table 5.8 summarizes all the message payloads necessary for computing the propagation delay. To evaluate the propagation delay, we assume that the link data rate is 6 Mbps.

Table 5.8: Average timings for calculating propagation delay

Scheme / **Message Size**	**EC-Schnorr SA-KMP**	**ECDSA PKR**	**ECDSA PKI**
Unsigned message (bytes)	69	-	-
Signed message (bytes)	125	125	125
Signed message with certificate (bytes)	-	-	251
CRL (Mbytes)	-	-	0.9

With the help of Figure 5.9 and the values in Table 5.7 and Table 5.8, we tabulate the total latency of the three schemes in Table 5.9. Results show that the latency involved in the exchange of certificates and downloading of CRL by the certificate-based PKI is several orders of the magnitude higher than that incurred in a V2V and V2I communication by the SA-KMP scheme. Moreover, the V2V communication latency by the SA-KMP scheme is lower than that by the PKR scheme even though an authentication is introduced against DoS attacks. Our scheme is more superior because 1) no certificates are sent in the message transmission and 2) the EC-Schnorr signing operation is faster than the ECDSA as it does not involve modular inversions. Our scheme is also able to support V2I communication with a low network latency of 4.488ms which is not addressed by the PKR scheme. These results show that the proposed SA-KMP scheme cannot introduce a very long delay time in acquiring public keys.

Table 5.9: Network latency of different schemes

Schemes in V2V and V2I/I2V	Network Latency
SA-KMP (V2V only)	12.309
SA-KMP (V2I/I2V)	4.488
Certificate-based with CRL downloading	1365.911
Certificate-based without CRL downloading	117.097
PKR (V2V only)	13.462

5.4.4 Scalability

We analyze the scalability of our scheme for an increasing number of users and compare our results to the certificate-based PKI scheme [29] and the PKR scheme [104]. The scalability is evaluated in terms of three aspects: the transmission overhead, storage overhead, and the latency.

In assessing the transmission overhead of our scheme, we consider two cases, namely the V2V and V2I communication. In the first instance, a vehicle has to perform two rounds of authentication before the actual communication takes place. The first authentication is to retrieve the public key of the target vehicle while the second set of authentication is to exchange keying materials for the establishment of pairwise keys. Each round of the authentication process consists of generating an EC-Schnorr signature for mitigating DoS attacks. On the other hand, only one round of authentication is needed in a V2I communication since they have the RSU-PFD and VPFD respectively which eliminates the need for requesting public key. Suppose the EC-Schnorr signature is 56 bytes based on a key length of 224 bits, the total transmission overhead generated by a vehicle in a V2V communication will be 112 bytes while the transmission overhead in a V2I communication is 56 bytes. In the PKI scheme, the transmission overhead is (56+126)=182 bytes as given in Table 5.5 whereas, the transmission overhead of the PKR scheme is 56 bytes. Table 5.10 summarizes the transmission overhead of all the three schemes based on a single user.

In terms of the storage overhead, we analyze the storage requirements of the vehicle and RSU separately in our scheme. We assume that the vehicle population increases from 100 to 1 million while the number of RSU in the network is fixed at 1000 regardless of the increase in the vehicle population. We assume that the public key size of an RSU and a

Table 5.10: Transmission overhead for different schemes for a single user

Communication Types \ Scheme	EC-Schnorr SA-KMP	ECDSA PKR	ECDSA PKI
V2V (bytes)	112	56	182
V2I (bytes)	56	56	182

Table 5.11: Storage overheard for different schemes in bytes (**B**)

No. of users	**No. of revoked certificates**	**PKI**	**SA-KMP**		**PKR**
		CRL_{size}	$VPFD_{size}$	$RSU-PFD_{size}$	$VPFD_{size}$
100	10	204 **B**	9700 **B**	37000 **B**	39.35555 **B**
1000	100	1014 **B**	97000 **B**	37000 **B**	68.3594 **B**
10000	1000	9114 **B**B	970000 **B**	37000 **B**	358.398 **B**
100000	10000	90114 **B**	9700000 **B**	37000 **B**	3258.79 **B**
1000000	100000	900114 **B**	97000000 **B**	37000 **B**	32262.7 **B**

vehicle is 33 bytes and 29 bytes respectively, and the size of the TA's ECDSA signature is 64 bytes. If 4 bytes are used to store the vehicle ID and assuming there are 100 vehicles in the network, the size of the VPFD held by each RSU in the SA-KMP scheme would be 100 x (29+4+64) bytes = 9700 bytes as shown in Table 5.11. On the other hand, the size of the RSU-PFD in the SA-KMP scheme remains constant at 1000 x (4+33) bytes = 37000 bytes. In the certificate-based PKI scheme, we assume that the storage overhead is due to the need to download and store the CRL list. So, we estimate the size of the CRL shown in Table 5.11 based on a revocation rate of 10% and assuming that the CRL header size is 50 bytes, the TA's signature on the CRL is 64 bytes and each revoked certificate is 9 bytes. For estimating the size of the public file directory *(denoted by VPFD)* in the PKR scheme, we assume a storage requirement of (33+4) bytes to store the RSU's public key and ID pair as well as a storage requirement of (29+4) bytes to store the vehicle's public key and ID pair, respectively. We then evaluate the total storage overhead based on 1000 RSU and for vehicles between 100 and 1 million.

Figure 5.10 shows a composite graph where the primary axis denotes the scalability in terms of the transmission overhead in logarithmic values while the secondary axis, also in logarithm scale, denotes the storage overhead requirements. From Figure 5.10, we observe

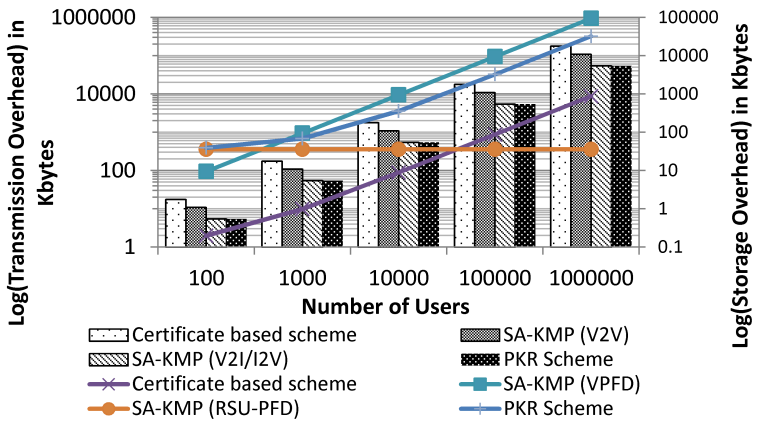

Figure 5.10: Scalability in terms of transmission overhead and storage overhead.

that the transmission overhead *(represented by column bar)* of our scheme in a V2I communication is the same as the PKR scheme. But, in the case of a V2V communication, our scheme incurs a higher transmission overhead compared to the PKR scheme due to the authentication mechanism introduced in the key agreement module which is used to mitigate flooding of invalid signatures. Nevertheless, the results suggest that the transmission overhead of our scheme is still lower than the classical PKI scheme. Suppose the typical data rate of a V-Mesh network is 27Mbps [4] and the transmission overhead in a V2V and V2I communication is 112 bytes and 56 bytes respectively, it would mean that our scheme is able to support between 30,000 and 60,000 users concurrently in a V2V and V2I communication respectively. In contrast, the certificate-based PKI scheme could only support up to 18,500 users since the transmission overhead is high at 182 bytes due to the sending of a certificate and an ECDSA signature for each communication. Even though the PKR scheme can support about 60,000 users which is similar to the SA-KMP in a V2I communication, we emphasize that the PKR scheme does not have protection against DoS attacks which is supported by our SA-KMP scheme.

In terms of the storage overhead *(represented by line graph)*, it is evident that VPFD in our scheme is higher than the PKR scheme as we are storing the TA's signature in the repository. However, the RSU-PFD which is maintained by each vehicle remains constant throughout because the RSU population rarely increases. Assuming the storage space to store an entry corresponding to a vehicle and RSU is 97 bytes and 37 bytes respectively, the total storage space required to store 1 million vehicles and 1000 RSUs by each RSU and vehicle will be about 92 Mbytes and 0.035 Mbytes respectively. This storage requirement is negligible given that modern-day OBU has a storage capacity of 8GB [132]. This observation supports our claim that SA-KMP scheme is highly scalable in terms of increasing

users. Although the certificate-based scheme has the lowest storage overhead, the main disadvantage of using certificates arises from the high latency involved in downloading the CRL list which is discussed next.

To evaluate the latency of our scheme as a function of increasing users, we develop a C program to measure the time taken by the SA-KMP scheme to search for a public key in the VPFD. We develop another C program to estimate the time required to check the validity of the certificate against the CRL of the certificate-based PKI scheme. The search operations by both programs are evaluated for different number of users ranging from 100 to 1 million and the average timing is taken over 100 simulation runs. Table 5.12 shows the simulation results for the various parameters under considerations. It is obvious that the searching time by the PKR scheme is the same as that of the SA-KMP scheme. Using the values in Table 5.7 and Table 5.8, we calculate the latency of each scheme by re-calculating the propagation delay and the processing delay as per Figure 5.9.

Table 5.12: Various parameters for calculating the latency

No. of users	**No. of revoked certificates**	**PKI**		**SA-KMP**	**PKR**
		CRL_{size}(bytes)	$T_{checkCRL}$ (ms)	T_{search} (ms)	T_{search} (ms)
100	10	204	0.023	0.000667	0.000667
1000	100	1014	0.071	0.00135	0.00135
10000	1000	9114	0.691	0.001663	0.001663
100000	10000	90114	7.910	0.003196	0.003196
1000000	100000	900114	125.987	0.008354	0.008354

Figure 5.11 shows the comparison of the latency between the three schemes for different number of users in the network. According to Figure 5.11, the latency to deploy the certificate-based PKI scheme increases sharply when the number of users is beyond 10,000. It is caused by the large CRL size, which results in a longer time to download and to search for a certificate. On the other hand, the latency in a V2V and V2I communication based on our scheme remains constant at around 12.5ms and 5ms irrespective of the number of users in the network. Furthermore, according to the left inset figure in Figure 5.11, the latency of the SA-KMP scheme is lower than that of the PKR scheme even though an authentication mechanism is introduced to mitigate the DoS attacks. These results conclude that our scheme is secure and highly scalable to cope with an increasing number

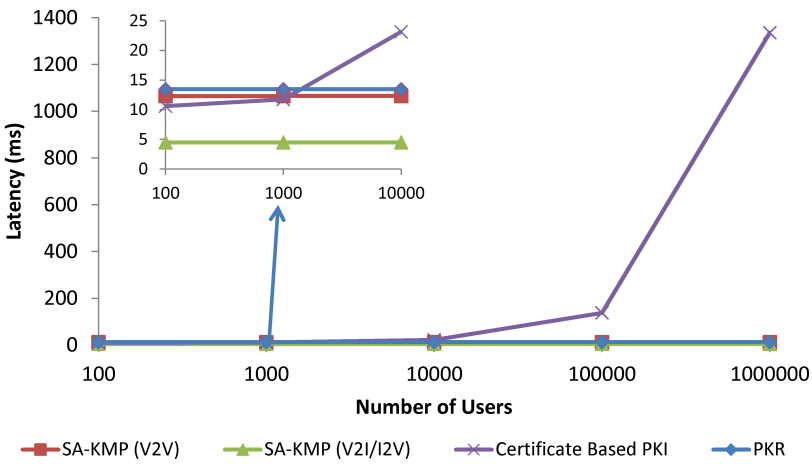

Figure 5.11: Scalability in terms of latency experienced by a user.

of users.

5.4.5 Key Generation Time

We develop a C program based on the key derivation module described in our scheme to calculate the time required to compute a pairwise key. First, we investigate the effects of varying the network size on the key generation time. After that, we study the relationship between the key generation time and the number of plane equations.

In the first experiment, we fix the number of plane equations N as 3 and we vary the modulus m operator in the equation (5.1) to be a prime number that approximates the network size between 0.5km and 10km. After that, we run the simulation 1000 times to measure the average key generation time and compare our results to the ECDH protocol [99] and the DH protocol [96]. Table 5.13 shows the various parameters for analyzing the key generation time of a function of network size.

Table 5.13: Simulation settings for the first experiment

Parameter	Value
No. of plane equations, N	3 (fixed)
Modulus operator, m	500m to 10000m
Simulation runs	1000
Benchmark	ECDH and DH

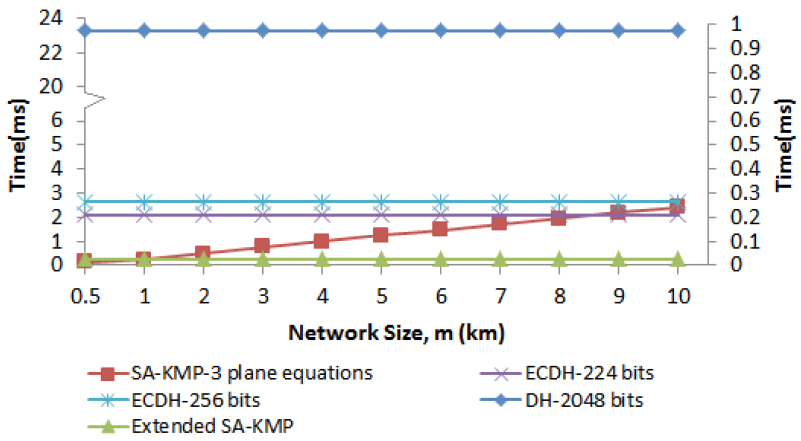

Figure 5.12: Key generation time as a function of network size.

Results in Figure 5.12 show that the average key generation time of the SA-KMP scheme is much lower than the DH protocol across the whole range of network sizes. Our scheme also outperforms the ECDH-256 and ECDH-224 protocols when the network size is smaller than 9km and 8km respectively. Furthermore, it can be seen in Figure 5.12 that the key generation time increases linearly with the network size. This is because the common keys are derived based on hashing the equation (5.4) a number of times according to the solution point. As the network size increases, it requires more hashing operations to determine the keys. To make our scheme more scalable in terms of the network size, we can reduce the number of hashing operations by taking the modulus (num) of the solution points. The modulus num can either be a fixed parameter pre-determined by the TA during the registration phase or it can be a parameter determined by the key requestor during the key agreement phase. The extended version of the SA-KMP scheme with reduced hashing operations is simulated using modulus $num = 100$ and the result is represented by the line with the green triangular markers in Figure 5.12. With reference to the secondary axis on Figure 5.12, the average key generation time for the extended version of the SA-KMP scheme is reduced and it remains constant at about 0.025ms throughout the various network sizes.

Next, we study the effects of increasing the number of plane equations N on the key generation time. In this experiment, the network size is fixed at 5km with modulus $m = 4999$ and the number of plane equations is varied from 3 to 10. The simulation is repeated for 1000 times to compute the average key generation time. Table 5.14 shows the various parameters for analyzing the key generation time of a function of plane equations.

Table 5.14: Simulation settings for second experiment

Parameter	Value
No. of plane equations, N	3 to 10
Modulus operator, m	4999m $\approx$ 5000m (fixed)
Simulation runs	1000

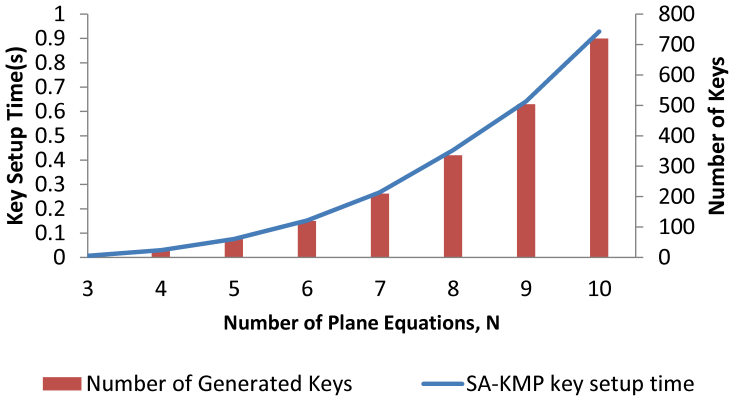

Figure 5.13: Key generation time as a function of plane equations.

Figure 5.13 shows that the key generation time increases exponentially with the number of plane equations. This is because there are $N(N-1)(N-2)$ groups of equations to solve and $N(N-1)(N-2)$ keys being generated for N plane equations. Although simulation result shows that it only takes about 0.93 seconds to generate 720 keys with 10 plane equations, the key generation time is going to scale up exponentially with respect to N which renders our scheme time-inefficient. To address this issue, we can randomly solve any combinations out of $\binom{N}{3}$ where N is the number of plane equations given. For example, if $N = 4$ and assuming all the plane equations issued by the TA are indexed, $\binom{N}{3}$ results in the following combinations where #number refers to the ID of the given plane equation: $\#1, \#2, \#3, \#1, \#2, \#4, \#1, \#3, \#4, \#2, \#3, \#4$. The key requestor can choose any sets out of the 4 combinations above to derive the common keys. If the key requestor chooses 3 sets from the possible combinations to solve, there will be 18 keys instead of 24. In this way, key generation time can be reduced. This makes our scheme highly configurable and flexible.

5.4.6 Computational Complexity

Next, we evaluate the computational complexity of the SA-KMP scheme by counting the number of CPU cycles it takes to execute in both V2V and V2I communication scenarios. In the V2V communication, both communicating parties need to acquire the public keys first before establishing the pairwise keys. Therefore, the complexity comes from the operations in the public key request module and the key derivation module. On the other hand, the complexity in a V2I communication is dominated by the operations of the key derivation module.

To determine the number of CPU cycles, we insert the read timestamp counter and processor (rdtscp ID) instruction in the C programs of the public key request module and the key derivation module. Both programs are executed for 1000 times on a 32 bit Debian Linux operating system running on an Intel Core i7-2620M processor workstation. Table 5.15 shows the average number of CPU cycles needed for both V2V and V2I communication scenarios.

Table 5.15: Number of cycles for C implementation of SA-KMP

	Number of CPU Cycles		**Total CPU Cycles**	**Computational Time (CT)**
Comm. Modes	Public Key Request	Key Derivation		
V2I/I2V	-	29.30 megacycles	29.30 megacycles	58.60 ms
V2V	42.54 megacycles	29.30 megacycles	70.84 megacycles	141.68 ms

Simulation results show that the V2V communications require 70.84 megacycles to complete which is about 57.4% higher than the V2I communications. The increase is due to the additional operations to acquire the public keys in the public key request module. Once we have obtained the total CPU cycles for each communication scenario, we can then convert them into actual computational time based on the different types of hardware specifications. The formula to convert CPU cycles to computational time is as follows:

$$CT = (clock\,cycles\,for\,a\,program)/(clock\,rate). \tag{5.8}$$

Suppose the OBU processor is 500MHz [132], the computational time for the V2I and V2V communication scenarios as calculated using (5.8) above will be 58.60 ms and 141.68 ms respectively as shown in the last column of Table 5.15. These results satisfies the latency requirements of typical V-Mesh network which further demonstrate the feasibility of our scheme in the real world settings.

5.5 Conclusion

In this paper, we have proposed the SA-KMP scheme that leverages on the PKR scheme and the 3D grid-key distribution scheme to reduce the latency and the complexity of a certificate-based PKI. The PKR scheme eliminates the complex certificate verification process while the 3D grid-key distribution scheme establishes symmetric keys to replace the expensive asymmetric cryptography.

Through numerical analysis, we have shown that SA-KMP scheme is more efficient and scalable than the certificate-based PKI scheme in terms of the network latency and transmission overhead, albeit a higher storage requirement. When compared to the PKR scheme, the transmission overhead and the storage overhead of SA-KMP is higher because of the extra authentication and the storing of TA's signature to combat DoS attacks and collusion attacks respectively. Nevertheless, the communication latency of the SA-KMP is still lower than the PKR scheme. In addition, despite the higher storage cost, the storage requirement is still well below the storage limit of a modern OBU with 8GB storage space [132]. Besides, we have shown that the SA-KMP scheme is highly configurable in terms of establishing pairwise keys. We have also demonstrated that under the DoS attacks, the SA-KMP scheme outperforms the ECDSA based schemes with a lower authentication delay. Lastly, the SA-KMP scheme is verified to be robust against a broad range of attacks using both formal verification like ProVerif and informal verifications.

In terms of future work, we intend to address privacy issues in our scheme and examine the case when RSUs are not fully trusted. We are also interested to extend our scheme to a scenario where RSUs are not pervasive in the region.

Chapter 6

Conclusions

With advances in wireless technology, many developments have been proposed to enhance the way people live, work and interact. The V-Mesh network can be seen as one of these new developments. It harnesses the power of communication, data, and transportation technologies to improve the road safety and to provide value-added services to enhance the comfort of road users. As demand for these application increases, the V-Mesh network has also become the target of many attacks. The attackers may behave selfishly by not forwarding the packets on behalf of others. The attackers may also behave maliciously by modifying packets during forwarding. In addition, the unique features of the V-Mesh network also pose challenges to undermine the effectiveness of a PKI. Thus, the main purpose of this research was to propose security measures to facilitate a trusted and secure communication in a vehicular environment. To this end, our security models stem from two aspects namely, the trust models and key management systems. In the following, we summarize our results and discuss our future work.

6.1 Summary

For a start, we studied the use of trust models to address selfish attacks such as blackhole and grayhole attacks. Based on our review, we have identified three key issues related to the trust model that require attention. First, in a trust model that gathers recommendation and direct trusts to assess the trustworthiness of a node, a malicious recommender may misreport recommendations by launching badmouthing and ballot-stuffing attacks in order to influence the detection of selfish node. Second, the prevalent mode of computing a trust rating using the overhearing technique is subject to exploitation whereby a node can control its transmission power to launch limited transmission power attacks. Third, in

addition to dropping packets during forwarding, a malicious node may also try to modify the packet intentionally before forwarding for its own selfish gains.

In response to the first issue, we proposed the DS-Trust model that draws inferences from human relationships in a social environment to handle the recommendation trusts. That is, the advice of others typically exhibit a certain degree of skepticism with respect to the actual events observed. Based on this idea, we use the principles of DST to model the deviation between the received recommendation trusts and the trust record held by oneself as uncertainty. Following this, the weight of the recommendation trusts is re-adjusted accordingly before they are combined with the direct trust record to form the final trust. We have shown via numerical analysis and simulations that DS-Trust model can mitigate badmouthing and ballot-stuffing attacks as well as improve the network throughput by avoiding selfish nodes during routing. More importantly, our approach contributes to the literature of trust systems with the following innovations. First, we avoid the challenging task of configuring an optimum deviation threshold to filter out the recommendations, which may sometimes result in counter-intuitive results. Second, we introduce uncertainty reasoning which makes the trust model more robust against a greater number of bad recommenders compared to existing trust approaches.

For the second and third issues, we proposed the MeTRO trust model that comprises of two techniques. The first technique is to enable a node to exploit the upstream monitoring reports of nodes that are two hops away to reinforce its own downstream monitoring observations. This helps to improve the overhearing process and mitigate limited transmission power attacks. The second technique introduces an authentication mechanism called the Merkle tree-based authentication to combat packet modification attacks. This mechanism first generates a Merkle commitment value for a message waiting to be sent and disseminates this value during the route discovery process. Subsequently, an authentication path corresponding to the sent packet is disseminated during packet forwarding to enable an intermediary along a multi-hop path to verify the integrity and authenticity of a packet. Simulation results indicate that MeTRO trust model is able to detect limited transmission power attacks, packet modification attacks including blackhole and grayhole attacks. Results also indicate that MeTRO is far more efficient than the ECDSA in terms of authentication delay. Interestingly, the time required to generate a Merkle commitment value and the time required to verify a data block increase exponentially with respect to the Merkle tree height. However, we note that this scalability problem can be resolved by fragmenting the original long message into more manageable fragments and construct multiple Merkle trees on these smaller fragments to keep the computation low. In essence, the

MeTRO trust model has the following contributions to the literature. First, to the best of our knowledge, our trust model is the first work that integrates a practical authentication mechanism into a routing protocol to detect packet modifications. Second, our trust model is the first that assess the trustworthiness of a node by considering both the forwarding behavior of a node and the integrity of forwarded packets.

The second part of the thesis examined the issues of using PKI and asymmetric cryptography to secure the V-Mesh network. The main issues of these solutions are the high latency and the poor scalability. This is largely due to the large CRL size and the resource-intensive asymmetric crypto-operations. Considering these issues, we developed a security framework called the SA-KMP that is efficient, scalable and secure. The efficiency gain is achieved by means of disseminating repositories containing the public keys of others to each network users. Thus, the exchange of certificates and the complex certificate management are eliminated. To alleviate the heavy costs of asymmetric cryptography, a novel key agreement protocol is designed where each entity negotiates common keys dynamically to secure communications symmetrically. Results indicate the SA-KMP has a lower transmission overhead and a lower latency compared to the certificate based PKI. Although the storage overhead due to the storing of public keys increases with the number of users in the network, we emphasize that the storage requirement is not a critical issue considering that a modern OBU boast a storage capacity of 8GB or more. Moreover, the main attraction of SA-KMP is the low latency which is a far more important consideration than the storage requirements. We have also analyzed the scalability of SA-KMP and show that transmission overhead and latency scale well with increasing users compared to the certificate based PKI. Although the key generation time in our key agreement protocol increases with the network size and the number of plane equations used, trade-offs could be made to balance between the security needs and the complexity costs. In terms of security, we provide formal and informal security validation to demonstrate that SA-KMP is secure against a wide range of attacks and further show that the key agreement protocol satisfies the authenticity and secrecy properties. To summarize, our study of SA-KMP contributes new understanding to the literature by constructing a scalable security framework that utilizes asymmetric as well as symmetric cryptography to support the latency sensitive applications in a V-Mesh network.

6.2 Recommendation for Future Work

6.2.1 Trust Models Extensions

Our current implementation of the DS-Trust model and the MeTRO trust model is based on a fixed trust threshold to detect if the downstream node is acting selfishly. As the applicate rate increases, the network contention will increase which causes packets to be dropped unintentionally. If the trust level of a node falls below the trust threshold, well-behaved nodes will be isolated leading to a high false detection rate. Consequently, fewer nodes are available to deliver the packets to the destination resulting in a low throughput. As a future work, we plan to estimate the wireless losses because of bad wireless channel quality or medium access collisions to set the trust threshold adaptively. This would allow us to better differentiate malicious packet drop from losses due to the unreliable wireless channel. This enhancement would allow our trust models to detect grayhole attackers with much higher accuracy and improve the PDR and throughput substantially.

Second, we plan to integrate our trust models into the position-based routing protocol to validate the applicability of our design. In our current implementation, we have used the AODV as a base to build our trust models which exhibit unsatisfactory performance in a V-Mesh network when (1) the number of nodes, (2) the mobility of nodes and (3) the number of flows is increased in the simulation. This restricts us to use a simplified model in order to establish a reasonable comparison. In contrast, the position-based routing exploits the position information of nodes to do routing which enables it to deal with frequent topology changes in a vehicular network [133]. By integrating our trust models onto a position-based routing, it enables us to validate other performance aspects of our trust design, especially the scalability in terms of increasing users and increasing mobility. Moreover, it allows us to identify further issues in our design for fine-tuning to make our trust models even more robust. More specifically, we plan to integrate our trust models into the Greedy Traffic Aware Routing (GyTAR) protocol [134] and compare the performance of DS-Trust and MeTRO trust model with a secure version of GyTAR protocol called the S-GyTAR [135].

Third, we plan to investigate Sybil attacks wherein attackers can fake multiple identities to subvert trust model. For instance, an attacker can create multiple identities to simulate different nodes in the network to issue dishonest recommendations. This attack can influence the trust rating of an honest node. Multiple identities in the network can also mislead the honest nodes to send packets to a non-existence node causing the packets

to never reach the intended destination. In the case of the MeTRO trust model where the upstream monitoring is used to strengthen the downstream observations, the effects of Sybil attacks will be more prevalent since the non-existence node would never send upstream reports to the evaluating node. Thus, the well-behaved nodes are blacklisted. Sybil attacks can also help a malicious attacker to evade detection. Thus, defense against Sybil attacks is critical to the overall operation of trust models.

6.2.2 SA-KMP Extensions

In the SA-KMP framework, pairwise and group keys in a V2V communication are established with the help of an RSU. In the former, a vehicle has to contact a RSU to retrieve the public key of the target vehicle before initiating the key agreement process. In the latter case, the RSU has to authenticate every vehicle in a group and distributes the group keys to all the authenticated vehicles in a group. As seen in our proposal, the RSU plays an important role in ensuring the key agreement process is executed properly. As a future work, we plan to extend our model to establish common keys without the help of the RSU. By doing so, it extends the applicability of our framework to suburban or rural scenario where the presence of RSU is not pervasive. Moreover, we can relax the assumption that RSU is fully trusted which in reality is more practical.

Privacy is another issue that has gained attention among the research community. In a V-Mesh network, the vehicles are publicly owned and are driven by users. They broadcast messages on a regular basis and assist other vehicles to forward the messages to the destination. Due to the broadcast nature of the wireless medium, any vehicles in the range of the transmitting vehicle are able to intercept and capture the messages for future analysis [122]. For example, the attackers can gather messages sent by the sender to trace its driving route and its identity. Subsequently, the attackers can reuse these messages to gain unauthorized access to other services. While privacy preservation is important, the authorities must also be able to track the misbehaving vehicle to determine responsibility. In our proposal, the establishment of symmetric keys for encryption has already helped to resolve the privacy issue due to the sharing of the same secret key. The ephemeral nature of the derived keys has also made the tracing of messages more difficult for the attackers. However, it is generally accepted that symmetric encryption is not able to provide sender accountability or traceability. Furthermore, in our key agreement protocol, each node is still required to exchange information such as its identifier and its GPS locations before the symmetric keys can be established. This requirement exposes the protocol to privacy attacks. Therefore, as a future work, we plan to extend the SA-KMP key agreement

protocol to achieve privacy and traceability requirements.

In addition to the above two issues, the proposed DoS mitigation technique based on the Schnorr signature scheme can also be exploited by external attacker to deplete the resources of the vehicle. This is because the vehicle must bear all the computational burden of verifying the signature from the RSU. If an external attacker succeeds in compromising a large number of vehicles, it may cause the network availability to deteriorate. An external attacker may also launch DDoS attacks on the RSU by simultaneously compromising multiple vehicles to send invalid request messages. As a future work, we plan to re-design the Schnorr signature scheme by considering the vehicle's trust level. If the vehicle identified by the RSU is reliable, the vehicle may skip the Schnorr-based authentication scheme to save energy. Therefore, even if an external attacker succeeds in compromising the vehicle, the scale and extent of the damage is reduced. To mitigate DDoS, we can implement a counter in the RSU to track the number of similar requests sent by each vehicle for a time period. If the frequency of request is very high, it is an indication of DDoS attacks.

Lastly, it would also be interesting to combine the work of MeTRO and SA-KMP as a starting ground to illustrate the workings of using non-cryptographic based solutions i.e. trust models and cryptography solutions to provide a comprehensive security framework. In this case, the trust models secure the network against internal selfish attacks to improve the network capacity. The SA-KMP aims to facilitate efficient communications to support the various diverse applications without exchanging certificates. The symmetric keys derived by SA-KMP aim to secure communication against both external and internal attacks between end-to-end nodes and providing confidentiality, authenticity and integrity properties. In fact, the symmetric keys can be used in the MeTRO trust model to realize the Merkle tree authentication mechanism. More specifically, the fragmented packets are encrypted using the derived symmetric keys to provide end-to-end security between the source and destination and to prevent the disclosure of information as the packets traversed along a multi-hop path. On the other hand, by the concept of Merkle authentication, the intermediate hops along the path are still able to verify the integrity and authenticity of the received packets efficiently without decrypting the packets.

List of Publications

Journal Papers:

[1] **H. C. Tan**, M. Ma, H. Labiod, P. H. J. Chong, and J. Zhang, "A Nonbiased Trust Model for Wireless Mesh Networks", *International Journal of Communication System*, 2016.

[2] **H. C. Tan**, M. Ma, H. Labiod, A. Boudguiga, J. Zhang, and P. H. J. Chong, "A Secure and Authenticated Key Management Protocol (SA-KMP) for Vehicular Networks", *IEEE Transactions on Vehicular Technology*, 65(12):9570-9584, 2016.

[3] **H. C. Tan**, M. Ma, H. Labiod, P. H. J. Chong, and J. Zhang, "A Merkle Tree-based Trust Scheme for Vehicular Mesh (V-Mesh) Networks," *Submitted to Elsevier journal of network and computer applications*

Conference Papers:

[1] **H. C. Tan**, M. Ma, H. Labiod, and P. H. J. Chong, "TEDS: A Trusted Entropy and Dempster Shafer Mechanism for Routing in Wireless Mesh Networks," *Proceedings of the Fourth International Conference on Mobile Services, Resources, and Users in MOBILITY*, pp. 12-18, 2014.

[2] **H. C. Tan**, J. Zhang, M. Ma, P. H. J. Chong, and H. Labiod, "Secure Public Key Regime (SPKR) in Vehicular Networks," *Proceedings of the 2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*, pp. 1-7, IEEE, 2015.

[3] J. Zhang, H. Labiod, M. Ren, **H. C. Tan**, "Cooperation Behavior of Vehicles in an Evolutionary Game for Information Dissemination," *Proceedings of the 2016 International Conference on Security of Smart Cities, Industrial Control System and Communications (SSIC)*, pp. 1-8, IEEE, 2016.

Bibliography

[1] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular Ad Hoc network," *Journal of network and computer applications*, vol. 37, pp. 380–392, 2014.

[2] H. Hartenstein and L. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Communications magazine*, vol. 46, no. 6, pp. 164–171, 2008.

[3] D. Jiang, V. Taliwal, A. Meier, W. Holfelder, and R. Herrtwich, "Design of 5.9 GHz DSRC-based vehicular safety communication," *IEEE Wireless Communications*, vol. 13, no. 5, pp. 36–43, 2006.

[4] Y. J. Li, "An overview of the DSRC/WAVE technology," in *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*. Springer, 2010, pp. 544–558.

[5] R. G. Engoulou, M. Bellaïche, S. Pierre, and A. Quintero, "VANET security surveys," *Computer Communications*, vol. 44, pp. 1–13, 2014.

[6] S. K. Bhoi and P. M. Khilar, "Vehicular communication: a survey," *IET Networks*, vol. 3, no. 3, pp. 204–217, 2014.

[7] L. Chen and C. Englund, "Cooperative ITS—EU standards to accelerate cooperative mobility," in *2014 International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE, 2014, pp. 681–686.

[8] C. Campolo, A. Molinaro, and R. Scopigno, *Vehicular ad hoc Networks: Standards, Solutions, and Research*. Springer, 2015.

[9] E. B. Hamida, H. Noura, and W. Znaidi, "Security of cooperative intelligent transport systems: Standards, threats analysis and cryptographic countermeasures," *Electronics*, vol. 4, no. 3, pp. 380–423, 2015.

[10] T. Taleb and A. Benslimane, "Design guidelines for a network architecture integrating vanet with 3g & beyond networks," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*. IEEE, 2010, pp. 1–5.

[11] S. Agrawal, N. Tyagi, and A. K. Misra, "Seamless VANET Connectivity through Heterogeneous Wireless Network on Rural Highways," in *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. ACM, 2016, p. 84.

[12] M. De Felice, F. Cuomo, A. Baiocchi, I. Turcanu, and S. Zennaro, "Traffic monitoring and incident detection using cellular and early stage VANET technology deployment," in *Proceedings of the First International Workshop on Internet of Vehicles and Vehicles of Internet*. ACM, 2016, pp. 1–6.

[13] A. A. Eltahir, R. A. Saeed, and R. A. Mokhtar, "Vehicular Communication and Cellular Network Integration: Gateway Selection Perspective," in *Computer and Communication Engineering (ICCCE), 2014 International Conference on*. IEEE, 2014, pp. 64–67.

[14] K. Zheng, Q. Zheng, P. Chatzimisios, W. Xiang, and Y. Zhou, "Heterogeneous Vehicular Networking: A Survey on Architecture, Challenges, and Solutions," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2377–2396, Fourthquarter 2015.

[15] L. Fuqiang and S. Lianhai, "Heterogeneous Vehicular Communication Architecture and Key Technologies," *ZTE Communications*, vol. 8, no. 4, pp. 39–44, Aug. 2010.

[16] M. Cesana, L. Fratta, M. Gerla, E. Giordano, and G. Pau, "C-VeT the UCLA campus vehicular testbed: Integration of VANET and Mesh networks," in *Wireless Conference (EW), 2010 European*. IEEE, 2010, pp. 689–695.

[17] I. F. Akyildiz and X. Wang, "A survey on wireless mesh networks," *IEEE Communications magazine*, vol. 43, no. 9, pp. S23–S30, 2005.

[18] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer networks*, vol. 47, no. 4, pp. 445–487, 2005.

[19] H. Soroush, N. Banerjee, A. Balasubramanian, M. D. Corner, B. N. Levine, and B. Lynn, "DOME: a diverse outdoor mobile testbed," in *Proceedings of the 1st ACM International Workshop on Hot Topics of Planet-Scale Mobility Measurements*. ACM, 2009, p. 2.

[20] C. Ameixieira, A. Cardote, F. Neves, R. Meireles, S. Sargento, L. Coelho, J. Afonso, B. Areias, E. Mota, R. Costa *et al.*, "Harbornet: a real-world testbed for vehicular networks," *IEEE Communications magazine*, vol. 52, no. 9, pp. 108–114, 2014.

[21] W. Liang, Z. Li, H. Zhang, S. Wang, and R. Bie, "Vehicular ad hoc networks: architectures, research issues, methodologies, challenges, and trends," *International Journal of Distributed Sensor Networks*, vol. 2015, p. 17, 2015.

[22] *ETSI TS 102 894-1: Intelligent Transport Systems (ITS); Users and applications requirements; Part 1: Facility layer structure, functional requirements and specifications V1.1.1*, Technical Specification, European Telecommunications Standards Institute, 2013.

[23] G. R. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke, "IEEE 802.11 s: the WLAN mesh standard," *IEEE Wireless Communications*, vol. 17, no. 1, pp. 104–111, 2010.

[24] R. A. Uzcátegui, A. J. De Sucre, and G. Acosta-Marum, "Wave: A tutorial," *IEEE Communications Magazine*, vol. 47, no. 5, pp. 126–133, 2009.

[25] M. Wolf and T. Gendrullis, "Design, implementation, and evaluation of a vehicular hardware security module," in *International Conference on Information Security and Cryptology*. Springer, 2011, pp. 302–318.

[26] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, and J.-P. Hubaux, "Secure vehicular communication systems: design and architecture," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 100–109, 2008.

[27] M.-F. Tsai, P.-C. Wang, C.-K. Shieh, W.-S. Hwang, N. Chilamkurti, S. Rho, and Y. S. Lee, "Improving positioning accuracy for VANET in real city environments," *The Journal of Supercomputing*, vol. 71, no. 6, pp. 1975–1995, 2015.

[28] C. Bergenhem, E. Hedin, and D. Skarin, "Vehicle-to-vehicle communication for a platooning system," *Procedia-Social and Behavioral Sciences*, vol. 48, pp. 1222–1233, 2012.

[29] *IEEE Standard for Wireless Access in Vehicular Environments Security Services for Applications and Management Messages*, IEEE Std 1609.2-2013 (Revision of IEEE Std 1609.2-2006), April 2013.

[30] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 4, pp. 584–616, 2011.

[31] M. Raya and J.-P. Hubaux, "Securing vehicular ad hoc networks," *Journal of Computer Security*, vol. 15, no. 1, pp. 39–68, 2007.

[32] A. M. Vegni, M. Biagi, and R. Cusani, *Smart vehicles, technologies and main applications in vehicular ad hoc networks.* INTECH Open Access Publisher, 2013.

[33] *ETSI TS 102 940: Intelligent Transport Systems (ITS); Security; ITS communications security architecture and security management V1.1.1*, Technical Specification, European Telecommunications Standards Institute, 2012.

[34] E. Schoch, F. Kargl, and M. Weber, "Communication patterns in vanets," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 119–125, 2008.

[35] J. Zhang, "Trust management for vanets: challenges, desired properties and future directions," *International Journal of Distributed Systems and Technologies (IJDST)*, vol. 3, no. 1, pp. 48–62, 2012.

[36] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking.* ACM, 2000, pp. 255–265.

[37] M. A. Hamid, M. S. Islam, and C. S. Hong, "Developing security solutions for wireless mesh enterprise networks," in *2008 IEEE Wireless Communications and Networking Conference.* IEEE, 2008, pp. 2549–2554.

[38] M. A. Hamid, M. Abdullah-Al-Wadud, C. S. Hong, O. Chae, and S. Lee, "A robust security scheme for wireless mesh enterprise networks," *annals of telecommunications-annales des télécommunications*, vol. 64, no. 5-6, pp. 401–413, 2009.

[39] M. N. Mejri, J. Ben-Othman, and M. Hamdi, "Survey on VANET security challenges and possible cryptographic solutions," *Vehicular Communications*, vol. 1, no. 2, pp. 53–66, 2014.

[40] M. S. Al-Kahtani, "Survey on security attacks in Vehicular Ad hoc Networks (VANETs)," in *Signal Processing and Communication Systems (ICSPCS), 2012 6th International Conference on.* IEEE, 2012, pp. 1–9.

[41] Y. Kim and I. Kim, "Security issues in vehicular networks," in *The International Conference on Information Networking 2013 (ICOIN)*. IEEE, 2013, pp. 468–472.

[42] P. Yi, Y. Wu, F. Zou, and N. Liu, "A survey on security in wireless mesh networks," *IETE Technical Review*, vol. 27, no. 1, pp. 6–14, 2010.

[43] S. Glass, M. Portmann, and V. Muthukkumarasamy, "Securing wireless mesh networks," *IEEE Internet Computing*, vol. 12, no. 4, pp. 30–36, 2008.

[44] J.-H. Cho, A. Swami, and R. Chen, "A survey on trust management for mobile ad hoc networks," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 4, pp. 562–583, 2011.

[45] A. Slagell, R. Bonilla, and W. Yurcik, "A survey of PKI components and scalability issues," in *2006 IEEE International Performance Computing and Communications Conference*. IEEE, 2006, pp. 10–pp.

[46] R. Housley, W. Polk, W. Ford, and D. Solo, "Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile," Internet Requests for Comments, RFC Editor, RFC 5280, May 2002. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5280.txt

[47] M. A. Javed, E. Ben Hamida, and W. Znaidi, "Security in intelligent transport systems for smart cities: From theory to practice," *Sensors*, vol. 16, no. 6, p. 879, 2016.

[48] C. Harsch, A. Festag, and P. Papadimitratos, "Secure position-based routing for VANETs," in *2007 IEEE 66th Vehicular Technology Conference*. IEEE, 2007, pp. 26–30.

[49] L. K. Qabajeh, M. L. M. Kiah, and M. M. Qabajeh, "Secure Unicast Position-based Routing Protocols for Ad-Hoc Networks," *Acta Polytechnica Hungarica*, vol. 8, no. 6, pp. 191–214, 2011.

[50] S. Buchegger and J.-Y. Le Boudec, "Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks," in *Parallel, Distributed and Network-based Processing, 2002. Proceedings. 10th Euromicro Workshop on*. IEEE, 2002, pp. 403–410.

[51] P. Michiardi and R. Molva, "Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Advanced communications and multimedia security*. Springer, 2002, pp. 107–121.

[52] X. Li, M. R. Lyu, and J. Liu, "A trust model based routing protocol for secure ad hoc networks," in *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, vol. 2. IEEE, 2004, pp. 1286–1295.

[53] S. Buchegger and J.-Y. Le Boudec, "A robust reputation system for peer-to-peer and mobile ad-hoc networks," in *P2PEcon 2004*, no. LCA-CONF-2004-009, 2004.

[54] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "Attacks on trust evaluation in distributed networks," in *2006 40th Annual Conference on Information Sciences and Systems*. IEEE, 2006, pp. 1461–1466.

[55] Y. L. Sun, W. Yu, Z. Han, and K. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 305–317, 2006.

[56] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "A Trust Evaluation Framework in Distributed Networks: Vulnerability Analysis and Defense Against Attacks." in *INFOCOM*, vol. 2006, 2006, pp. 1–13.

[57] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 4, no. 3, p. 15, 2008.

[58] Y. Sun, Z. Han, and K. R. Liu, "Defense of trust management vulnerabilities in distributed networks," *IEEE Communications Magazine*, vol. 46, no. 2, pp. 112–119, 2008.

[59] H. Hu, R. Lu, and Z. Zhang, "VTrust: A Robust Trust Framework for Relay Selection in Hybrid Vehicular Communications," in *2015 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2015, pp. 1–6.

[60] E. Hernandez-Orallo, M. D. S. Olmos, J.-C. Cano, C. T. Calafate, and P. Manzoni, "CoCoWa: a collaborative contact-based watchdog for detecting selfish nodes," *IEEE Transactions on Mobile Computing*, vol. 14, no. 6, pp. 1162–1175, 2015.

[61] A. El Khatib, A. Mourad, H. Otrok, O. A. Wahab, and J. Bentahar, "A Cooperative Detection Model Based on Artificial Neural Network for VANET QoS-OLSR Protocol," in *2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*. IEEE, 2015, pp. 1–5.

[62] D. M. Shila, Y. Cheng, and T. Anjali, "Mitigating selective forwarding attacks with a channel-aware approach in WMNs," *IEEE transactions on wireless communications*, vol. 9, no. 5, pp. 1661–1675, 2010.

[63] A. A. Pirzada and C. McDonald, "Establishing trust in pure ad-hoc networks," in *Proceedings of the 27th Australasian conference on Computer science-Volume 26*. Australian Computer Society, Inc., 2004, pp. 47–54.

[64] A. A. Pirzada, C. McDonald, and A. Datta, "Performance comparison of trust-based reactive routing protocols," *IEEE transactions on Mobile computing*, vol. 5, no. 6, pp. 695–710, 2006.

[65] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the CONFIDANT protocol," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*. ACM, 2002, pp. 226–236.

[66] O. A. Wahab, H. Otrok, and A. Mourad, "A cooperative watchdog model based on Dempster–Shafer for detecting misbehaving vehicles," *Computer Communications*, vol. 41, pp. 43–54, 2014.

[67] A. Jesudoss, S. K. Raja, and A. Sulaiman, "Stimulating truth-telling and cooperation among nodes in VANETs through payment and punishment scheme," *Ad Hoc Networks*, vol. 24, pp. 250–263, 2015.

[68] Q. He, D. Wu, and P. Khosla, "SORI: a secure and objective reputation-based incentive scheme for ad-hoc networks," in *Wireless communications and networking conference, 2004. WCNC. 2004 IEEE*, vol. 2. IEEE, 2004, pp. 825–830.

[69] J. O. Berger, *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013.

[70] S. Buchegger and J.-Y. Le Boudec, "A robust reputation system for mobile ad-hoc networks," Tech. Rep., 2003.

[71] R. Chen, J. Guo, F. Bao, and J.-H. Cho, "Trust management in mobile ad hoc networks for bias minimization and application performance maximization," *Ad Hoc Networks*, vol. 19, pp. 59–74, 2014.

[72] L. Ruidong, L. Jie, and H. Asaeda, "A Hybrid Trust Management Framework for Wireless Sensor and Actuator Networks in Cyber-Physical Systems," *IEICE TRANSACTIONS on Information and Systems*, vol. 97, no. 10, pp. 2586–2596, 2014.

[73] A. Jsang and R. Ismail, "The beta reputation system," in *Proceedings of the 15th bled electronic commerce conference*, vol. 5, 2002, pp. 2502–2511.

[74] A. Jøsang, "Artificial reasoning with subjective logic," in *Proceedings of the second Australian workshop on commonsense reasoning*, vol. 48. Citeseer, 1997, p. 34.

[75] A. Jøsang, E. Gray, and M. Kinateder, "Simplification and analysis of transitive trust networks," *Web Intelligence and Agent Systems: An International Journal*, vol. 4, no. 2, pp. 139–161, 2006.

[76] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *Proceedings of the 29th Australasian Computer Science Conference-Volume 48*. Australian Computer Society, Inc., 2006, pp. 85–94.

[77] K. Kane and J. C. Browne, "Using uncertainty in reputation methods to enforce cooperation in ad-hoc networks," in *Proceedings of the 5th ACM workshop on Wireless security*. ACM, 2006, pp. 105–113.

[78] Y. Liu, K. Li, Y. Jin, Y. Zhang, and W. Qu, "A novel reputation computation model based on subjective logic for mobile ad hoc networks," *Future Generation Computer Systems*, vol. 27, no. 5, pp. 547–554, 2011.

[79] H. Lin, J. Ma, J. Hu, and K. Yang, "PA-SHWMP: a privacy-aware secure hybrid wireless mesh protocol for IEEE 802.11 s wireless mesh networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, pp. 1–16, 2012.

[80] S. Dietzel, R. van der Heijden, H. Decke, and F. Kargl, "A flexible, subjective logic-based framework for misbehavior detection in V2V networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*. IEEE, 2014, pp. 1–6.

[81] Y. Wang, R. Chen, and J.-H. Cho, "Trust-based Service Management of Mobile Devices in Ad Hoc Networks," unpublished.

[82] Y. Wang, Y.-C. Lu, I.-R. Chen, J.-H. Cho, and A. Swami, "LogitTrust: A logit regression-based trust model for mobile ad hoc networks," in *6th ASE International Conference on Privacy, Security, Risk and Trust*. Boston, MA, 2014.

[83] Z. Ullah, M. H. Islam, A. A. Khan, and S. Sarwar, "Filtering dishonest trust recommendations in trust management systems in mobile ad hoc networks," *International*

Journal of Communication Networks and Information Security, vol. 8, no. 1, p. 18, 2016.

[84] A. Arning, R. Agrawal, and P. Raghavan, "A linear method for deviation detection in large databases." in *KDD*, 1996, pp. 164–169.

[85] K. Plößl and H. Federrath, "A privacy aware and efficient security infrastructure for vehicular ad hoc networks," *Computer Standards & Interfaces*, vol. 30, no. 6, pp. 390–397, 2008.

[86] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM conference on Computer and communications security.* ACM, 2002, pp. 41–47.

[87] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Security and Privacy, 2003. Proceedings. 2003 Symposium on.* IEEE, 2003, pp. 197–213.

[88] J. Almeida, S. Shintre, M. Boban, and J. Barros, "Probabilistic key distribution in vehicular networks with infrastructure support," in *Global Communications Conference (GLOBECOM), 2012 IEEE.* IEEE, 2012, pp. 973–978.

[89] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Annual International Cryptology Conference.* Springer, 1992, pp. 471–486.

[90] W. Zhang, M. Tran, S. Zhu, and G. Cao, "A random perturbation-based scheme for pairwise key establishment in sensor networks," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing.* ACM, 2007, pp. 90–99.

[91] D. A. Don, V. Pandit, and D. P. Agrawal, "Multivariate symmetric polynomial based group key management for vehicular ad hoc networks," in *2012 IEEE International Conference on Communications (ICC).* IEEE, 2012, pp. 7172–7176.

[92] R. Blom, "An optimal class of symmetric key generation systems," in *Workshop on the Theory and Application of of Cryptographic Techniques.* Springer, 1984, pp. 335–338.

[93] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *ACM Transactions on Information and System Security (TISSEC)*, vol. 8, no. 2, pp. 228–258, 2005.

[94] Y. Zhang, L. Xu, Y. Xiang, and X. Huang, "A Matrix-Based Pairwise Key Establishment Scheme for Wireless Mesh Networks Using Pre Deployment Knowledge," *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 2, pp. 331–340, 2013.

[95] ——, "Matrix-based pairwise key establishment in wireless mesh networks using deployment knowledge," in *2013 IEEE International Conference on Communications (ICC)*. IEEE, 2013, pp. 1604–1608.

[96] W. Hsin-Te, W.-S. Li, S. Tung-Shih, and W.-S. Hsiehz, "A novel RSU-based message authentication scheme for VANET," in *2010 Fifth International Conference on Systems and Networks Communications*. IEEE, 2010, pp. 111–116.

[97] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[98] D. R. Stinson, *Cryptography: theory and practice*. CRC press, 2005.

[99] Q. Niu, "ECDH-based scalable distributed key management scheme for secure group communication," *Journal of Computers*, vol. 9, no. 1, pp. 153–160, 2014.

[100] D. Brown, "Standards for efficient cryptography, SEC 1: elliptic curve cryptography," *Released Standard Version*, vol. 1, 2009.

[101] G. Li and D. J. Wheeler, "A matrix key-distribution scheme," *Journal of cryptology*, vol. 2, no. 1, pp. 51–59, 1990.

[102] B. Zan, M. Gruteser, and F. Hu, "Key agreement algorithms for vehicular communication networks based on reciprocity and diversity theorems," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 8, pp. 4020–4027, 2013.

[103] J.-F. Raymond and A. Stiglic, "Security issues in the Diffie-Hellman key agreement protocol," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–17, 2000.

[104] P.-Y. Shen, V. Liu, M. Tang, and W. Caelli, "An efficient public key management system: an application in vehicular ad hoc networks," in *Pacific Asia Conference on Information Systems (PACIS)*. AIS Electronic Library (AISeL), 2011, p. 175.

[105] M. Al-Qutayri, C. Yeun, and F. Al-Hawi, *Security and privacy of intelligent VANETs*. INTECH Open Access Publisher, 2010.

[106] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Workshop on the Theory and Application of Cryptographic Techniques*. Springer, 1984, pp. 47–53.

[107] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM journal on computing*, vol. 32, no. 3, pp. 586–615, 2003.

[108] D. He, S. Zeadally, B. Xu, and X. Huang, "An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2681–2691, 2015.

[109] C.-P. Schnorr, "Efficient signature generation by smart cards," *Journal of cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[110] A. Hesham, A. Abdel-Hamid, and M. A. El-Nasr, "A dynamic key distribution protocol for PKI-based VANETs," in *Wireless Days (WD), 2011 IFIP*. IEEE, 2011, pp. 1–3.

[111] A. H. Salem, A. Abdel-Hamid, and M. A. El-Nasr, "The Case for Dynamic Key Distribution for PKI-Based VANETs," *arXiv preprint arXiv:1605.04696*, 2016.

[112] A. Wasef and X. Shen, "EMAP: Expedite message authentication protocol for vehicular ad hoc networks," *IEEE transactions on Mobile Computing*, vol. 12, no. 1, pp. 78–89, 2013.

[113] G. Shafer *et al.*, *A mathematical theory of evidence*. Princeton university press Princeton, 1976, vol. 1.

[114] Z. Li and C. T. Chigan, "On joint privacy and reputation assurance for vehicular ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 10, pp. 2334–2344, 2014.

[115] S.-J. Horng, S.-F. Tzeng, Y. Pan, P. Fan, X. Wang, T. Li, and M. K. Khan, "b-SPECS+: Batch verification for secure pseudonymous authentication in VANET," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1860–1875, 2013.

[116] Y. Hao, Y. Cheng, C. Zhou, and W. Song, "A distributed key management framework with cooperative message authentication in VANETs," *IEEE Journal on selected areas in communications*, vol. 29, no. 3, pp. 616–629, 2011.

[117] A. P. Dempster, "Upper and lower probabilities induced by a multivalued mapping," *The annals of mathematical statistics*, pp. 325–339, 1967.

[118] C. Perkins, E. Belding-Royer, S. Das *et al.*, "RFC 3561-ad hoc on-demand distance vector (AODV) routing," *Internet RFCs*, pp. 1–38, 2003.

[119] "NS-3 network simulator homepage," http://www.nsnam.org Retrieved: January 2016.

[120] "Vehicular mobility trace of the city of Cologne, Germany,," http://kolntrace.project.citi-lab.fr/. Retrieved: December 2016.

[121] R. C. Merkle, "Protocols for Public Key Cryptosystems." in *IEEE Symposium on Security and privacy*, vol. 122, 1980.

[122] J. Petit, F. Schaub, M. Feiri, and F. Kargl, "Pseudonym schemes in vehicular networks: A survey," *IEEE communications surveys & tutorials*, vol. 17, no. 1, pp. 228–255, 2015.

[123] J. Lin, "Divergence measures based on the Shannon entropy," *IEEE Transactions on Information theory*, vol. 37, no. 1, pp. 145–151, 1991.

[124] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "BonnMotion: a mobility scenario generation and analysis tool," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010, p. 51.

[125] A. Wasef, R. Lu, X. Lin, and X. Shen, "Complementing public key infrastructure to secure vehicular ad hoc networks [security and privacy in emerging wireless networks]," *IEEE Wireless Communications*, vol. 17, no. 5, pp. 22–28, 2010.

[126] Y. Seurin, "On the exact security of schnorr-type signatures in the random oracle model," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2012, pp. 554–571.

[127] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.

[128] B. Blanchet, "Automatic verification of correspondences for security protocols," *Journal of Computer Security*, vol. 17, no. 4, pp. 363–434, 2009.

[129] J.-P. Hubaux, S. Capkun, and J. Luo, "The security and privacy of smart vehicles," *IEEE Security & Privacy Magazine*, vol. 2, no. LCA-ARTICLE-2004-007, pp. 49–55, 2004.

[130] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on information theory*, vol. 29, no. 2, pp. 198–208, 1983.

[131] M. Ullmann, C. Wieschebrink, and D. Kügler, "Public key infrastructure and crypto agility concept for intelligent transportation systems," in *VEHICULAR 2015, The Fourth International Conference on Advances in Vehicular Systems, Technologies and Applications*, 2015, Conference Proceedings, pp. 14–19.

[132] "Savari Homepage - MobiWave OBU," http://www.savarinetworks.com/products/mobiwave/ Retrieved: January 2016.

[133] K. Katsaros, M. Dianati, and K. Roscher, "A position-based routing module for simulation of vanets in ns-3," in *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2012, pp. 345–352.

[134] M. Jerbi, S.-M. Senouci, T. Rasheed, and Y. Ghamri-Doudane, "Towards efficient geographic routing in urban vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 9, pp. 5048–5059, 2009.

[135] T. Bouali, E.-H. Aglzim, and S.-M. Senouci, "A secure intersection-based routing protocol for data collection in urban vehicular networks," in *2014 IEEE Global Communications Conference*. IEEE, 2014, pp. 82–87.

Vers des communications de confiance et sécurisées dans un environnement véhiculaire

Heng Chuan Tan

RESUME : Le routage et la gestion des clés sont les plus grands défis dans les réseaux de véhicules. Un comportement de routage inapproprié peut affecter l'efficacité des communications et affecter la livraison des applications liées à la sécurité. D'autre part, la gestion des clés, en particulier en raison de l'utilisation de la gestion des certificats PKI, peut entraîner une latence élevée, ce qui peut ne pas convenir à de nombreuses applications critiques. Pour cette raison, nous proposons deux modèles de confiance pour aider le protocole de routage à sélectionner un chemin de bout en bout sécurisé pour le transfert. Le premier modèle se concentre sur la détection de nœuds égoïstes, y compris les attaques basées sur la réputation, conçues pour compromettre la «vraie» réputation d'un nœud. Le second modèle est destiné à détecter les redirecteurs qui modifient le contenu d'un paquet avant la retransmission. Dans la gestion des clés, nous avons développé un système de gestion des clés d'authentification et de sécurité (SA-KMP) qui utilise une cryptographie symétrique pour protéger la communication, y compris l'élimination des certificats pendant la communication pour réduire les retards liés à l'infrastructure PKI.

MOTS-CLEFS : Règle de combinaison de Dempster, fusion d'informations, attaques par paquets, modèle de confiance basé sur la recommandation, attaques basées sur la réputation, attaques à transmission limitée, mécanisme d'authentification Merkle tree, VANET, WMN, PKI sans certi cat, cryptosystèmes hybrides, Proverif, grille 3D accord de clé basé sur

ABSTRACT : Routing and key management are the biggest challenges in vehicular networks. Inappropriate routing behaviour may affect the effectiveness of communications and affect the delivery of safety-related applications. On the other hand, key management, especially due to the use of PKI certificate management, can lead to high latency, which may not be suitable for many time-critical applications. For this reason, we propose two trust models to assist the routing protocol in selecting a secure end-to-end path for forwarding. The first model focusses on detecting selfish nodes, including reputation-based attacks, designed to compromise the "true" reputation of a node. The second model is intended to detect forwarders that modify the contents of a packet before retransmission. In key management, we have developed a Secure and Authentication Key Management Protocol (SA-KMP) scheme that uses symmetric cryptography to protect communication, including eliminating certificates during communication to reduce PKI-related delays.

KEY-WORDS : Dempster's rule of combination, information fusion, packet dropping attacks, recommendation-based trust model, reputation-based attacks, limited transmission power attacks, Merkle tree authentication mechanism, VANET, WMN, certificate-less PKI, hybrid cryptosystems, Proverif, 3D grid-based key agreement

TELECOM ParisTech

EDITE

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY