Collaborative security for the internet of things

Yosra Ben Saied

▶ **To cite this version:**

Yosra Ben Saied. Collaborative security for the internet of things. Economics and Finance. Institut National des Télécommunications, 2013. English. NNT : 2013TELE0013 . tel-00879790

HAL Id: tel-00879790
https://theses.hal.science/tel-00879790

Submitted on 4 Nov 2013

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.

THESE DE DOCTORAT CONJOINT TELECOM SUDPARIS et L'UNIVERSITE PIERRE ET MARIE CURIE

Spécialité : Télécommunications

Ecole doctorale : Informatique, Télécommunications et Electronique de Paris

Présentée par

Yosra Ben Saied

Pour obtenir le grade de
DOCTEUR DE TELECOM SUDPARIS

Sécurité Collaborative pour l'Internet des Objets

Soutenue le 14/06/2013 devant le jury composé de :

Pr. Djamal Zeghlache	Professeur, Directeur de Département à Télécom SudParis	Directeur de Thèse
Alexis Olivereau	Ingénieur-chercheur au CEA LIST	Encadrant
Pr. Maryline Laurent	Professeur à Télécom SudParis	Co-encadrante
Pr. Isabelle Christment	Professeur à Computer Science, TELECOM Nancy, Université de Lorraine	Rapporteur
Pr. Yacine Challal	Maitre de conférences à Université de Technologie de Compiègne	Rapporteur
Pr. Guy Pujolle	Professeur à l'Université Paris 6	Examinateur
Dr. Olivier Heen	Chercheur à Technicolor	Examinateur

Thèse n° 2013TELE0013

THESE DE DOCTORAT CONJOINT TELECOM SUDPARIS et L'UNIVERSITE PIERRE ET MARIE CURIE

Specialisation: Telecommunications

Doctoral School: Information, Telecommunications and Electronics of Paris.

Presented by

Yosra Ben Saied

For the degree of
DOCTOR OF TELECOM SUDPARIS

Collaborative Security for the Internet of Things

Defended on 14/06/2013 in front of a jury composed of:

Pr. Djamal Zeghlache	Professor, Head of Department at Telecom SudParis	Thesis director
Alexis Olivereau	Research Engineer at CEA LIST	Advisor
Pr. Maryline Laurent	Professor at Telecom SudParis	Co-advisor
Pr. Isabelle Christment	Professor at Computer Science, TELECOM Nancy, University of Lorraine	Reviewer
Pr. Yacine Challal	Associate Professor at University of Technology of Compiègne	Reviewer
Pr. Guy Pujolle	Professor at University Paris 6	Examiner
Dr. Olivier Heen	Senior Scientist at Technicolor	Examiner

Thèse n° 2013TELE0013

ABSTRACT

This thesis addresses new security challenges in the Internet of Things (IoT). The current transition from legacy Internet to Internet of Things leads to multiple changes in its communication paradigms. Wireless sensor networks (WSNs) initiated this transition by introducing unattended wireless topologies, mostly made of resource constrained nodes, in which radio spectrum therefore ceased to be the only resource worthy of optimization. Today's Machine to Machine (M2M) and Internet of Things architectures further accentuated this trend, not only by involving wider architectures but also by adding heterogeneity, resource capabilities inconstancy and autonomy to once uniform and deterministic systems.

The heterogeneous nature of IoT communications and imbalance in resources capabilities between IoT entities make it challenging to provide the required end-to-end secured connections. Unlike Internet servers, most of IoT components are characterized by low capabilities in terms of both energy and computing resources, and thus, are unable to support complex security schemes. The setup of a secure end-to-end communication channel requires the establishment of a common secret key between both peers, which would be negotiated relying on standard security key exchange protocols such as Transport Layer Security (TLS) Handshake or Internet Key Exchange (IKE). Nevertheless, a direct use of existing key establishment protocols to initiate connections between two IoT entities may be impractical unless both endpoints be able to run the required (expensive) cryptographic primitives– thus leaving aside a whole class of resource-constrained devices. The issue of adapting existing security protocols to fulfil these new challenges has recently been raised in the international research community but the first proposed solutions failed to satisfy the needs of resource-constrained nodes.

In this thesis, we propose novel collaborative approaches for key establishment designed to reduce the requirements of existing security protocols, in order to be supported by resource-constrained devices. We particularly retained TLS handshake, Internet key Exchange and HIP BEX protocols as the best keying candidates fitting the end-to-end security requirements of the IoT. Then we redesigned them so that the constrained peer may delegate its heavy cryptographic load to less constrained nodes in neighbourhood exploiting the spatial heterogeneity of IoT nodes. Formal security verifications and performance analyses were also conducted to ensure the security effectiveness and energy efficiency of our collaborative protocols.

However, allowing collaboration between nodes may open the way to a new class of threats, known as internal attacks that conventional cryptographic mechanisms fail to deal with. This introduces the concept of trustworthiness within a collaborative group. The trustworthiness level of a node has to be assessed by a dedicated security mechanism known as a trust management system. This system aims to track nodes behaviours to detect untrustworthy elements and select reliable ones for collaborative services assistance. In turn, a trust management system is instantiated on a collaborative basis, wherein multiple nodes share their evidences about one another's trustworthiness. Based on an extensive analysis of prior trust management systems, we have identified a set of best practices that provided us guidance to design an effective trust management system for our collaborative keying protocols. This effectiveness was assessed by considering how the trust management system could fulfil specific requirements of our proposed approaches for key establishment in the context of the IoT. Performance analysis results show the proper functioning and effectiveness of the proposed system as compared with its counterparts that exist in the literature.

KEY WORDS:
Internet of Things, Wireless Sensor Networks, M2M, heterogeneity, resource constraints, end-to-end security, key establishment, energy efficiency, collaboration, internal attacks, trust.

RESUME

Cette thèse aborde des nouveaux défis de sécurité dans l'Internet des Objets (IdO). La transition actuelle de l'Internet classique vers l'Internet des Objets conduit à de nombreux changements dans les modèles de communications sous-jacents. Les réseaux de capteurs sans fil ont initié cette transition en introduisant des topologies sans fil, sans opérateur humain, et principalement composées de nœuds à ressources limitées. Aujourd'hui, les architectures Machine à Machine (M2M) et Internet des Objets accentuent cette évolution, non seulement en mettant en œuvre des ensembles de nœuds plus importants, mais aussi en intégrant une plus grande autonomie et une plus grande hétérogénéité entre les nœuds (disparates en particulier du point de vue de leurs contraintes en ressources) à des systèmes jusqu'alors déterministes et uniformes.

La nature hétérogène des communications de l'IdO et le déséquilibre entre les capacités des entités communicantes qui le constituent rendent difficile l'établissement de connexions sécurisées de bout en bout. Contrairement aux nœuds de l'Internet traditionnel, la plupart des composants de l'Internet des Objets sont en effet caractérisés par de faibles capacités en termes d'énergie et de puissance calcul. Par conséquent, ils ne sont pas en mesure de supporter des systèmes de sécurité complexes. En particulier, la mise en place d'un canal de communication sécurisé de bout en bout nécessite l'établissement d'une clé secrète commune entre les deux nœuds souhaitant communiquer, qui sera négociée en s'appuyant sur un protocole d'échange de clés tels que le Transport Layer Security (TLS) Handshake ou l'Internet Key Exchange (IKE). Or, une utilisation directe de ces protocoles pour établir des connexions sécurisées entre deux entités de l'IdO peut être difficile en raison de l'écart technologique entre celles-ci et des incohérences qui en résultent sur le plan des primitives cryptographiques supportées. Le sujet de l'adaptation des protocoles de sécurité existants pour répondre à ces nouveaux défis a récemment été soulevé dans la communauté scientifique. Cependant, les premières solutions proposées n'ont pas réussi à répondre aux besoins des nœuds à ressources limitées.

Dans cette thèse, nous proposons de nouvelles approches collaboratives pour l'établissement de clés, dans le but de réduire les exigences des protocoles de sécurité existants, afin que ceux-ci puissent être mis en œuvre par des nœuds à ressources limitées. Nous avons particulièrement retenu les protocoles TLS Handshake, IKE et HIP BEX comme les meilleurs candidats correspondant aux exigences de sécurité de bout en bout pour l'IdO. Puis nous les avons modifiés de sorte que le nœud contraint en énergie puisse déléguer les opérations cryptographiques couteuses à un ensemble de nœuds au voisinage, tirant ainsi avantage de l'hétérogénéité spatiale qui caractérise l'IdO. Nous avons entrepris des vérifications formelles de sécurité et des analyses de performance qui prouvent la sureté et l'efficacité énergétique des protocoles collaboratifs proposés.

Dans une deuxième partie, nous avons porté notre attention sur une classe d'attaques internes que la collaboration entre les nœuds peut induire et que les mécanismes cryptographiques classiques, tels que la signature et le chiffrement, s'avèrent impuissants à contrer. Cela nous a amené à introduire la notion de confiance au sein d'un groupe collaboratif. Le niveau de fiabilité d'un nœud est évalué par un mécanisme de sécurité dédié, connu sous le nom de système de gestion de confiance. Ce système est lui aussi instancié sur une base collaborative, dans laquelle plusieurs nœuds partagent leurs témoignages respectifs au sujet de la fiabilité des autres nœuds. En nous appuyant sur une analyse approfondie des systèmes de gestion de confiance existants et des contraintes de l'IoD, nous avons conçu un système de gestion de confiance efficace pour nos protocoles collaboratifs. Cette efficacité a été évaluée en tenant compte de la façon dont le système de gestion de la confiance répond aux exigences spécifiques à nos approches proposées pour l'établissement de clés dans le contexte de l'IdO. Les résultats des analyses de performance que nous avons menées démontrent le bon fonctionnement du système proposé et une efficacité accrue par rapport à la littérature.

ACKNOWLEDGEMENTS

The submission of this thesis brings to an end of a wonderful period, of almost three years, in which I was a PhD student at the Communicating Systems Laboratory in the French Atomic Energy Commission. On my way to complete this thesis, I have experienced many happy moments, as well as hurdles. I would like to thank those who gave me the strength and courage to continue and press forward.

My greatest appreciation and gratitude go first to my advisor Alexis Olivereau for his trust, encouragement and support. Alexis was earlier my supervisor during my master internship. He was the reason why I decided to go to pursue a career in research and start a thesis. During almost four years, he has been a role model for me, with his all-encompassing knowledge, inquisitive mind, uncompromising integrity, and enviable ability to conduct many diverse researches in parallel. I would like to thank him for guiding me through this important period of my life.

My gratitude extends to my thesis director Prof. Djamal Zeghlache for his scientific advice, vast knowledge and insightful suggestions during all phases of my thesis. I have had a pleasure and a great honor to be his PhD student. He gives the example to follow to be an effective scientist in the future, combining impressive expertise with constant humility. I would like to thank him for his support and belief in my work.

I am deeply grateful also to my co-advisor, Prof. Maryline Laurent for giving so generously of her time and providing useful advices. She was involved in the supervision of my thesis from the second year on and was supportive and influential to my research in a number of ways. It was a great privilege to learn from her.

I would like also to say thank you to Christophe Janneteau, the Lab Manager, and all my current and past colleagues at LSC laboratory. They made my time at CEA an experience that far surpassed my every expectation. The environment they created is probably the best a PhD student could ever hope for.

I would like to thank all my family for their unconditional love, care and encouragement. I will be forever thankful to my father for instilling in me the love of learning and the continuous desire for more knowledge. I thank my mother for her endless support, her absolute faith in me and for reminding me the important things in life. I owe her much more then I would ever be able to express.

I dedicate this thesis to the memory of my cherished grandmother who passed away as this final version of my thesis was nearing completion.

Last and not least, I would like to thank GOD for making 2013 the year of more. Thank you for your blessings. Thank you for your love. Thank you for my life.

Yosra BEN SAIED

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

6LoWPAN	IPv6 Low power Wireless Personal Area Networks
AF	Amplify and Forward
AODV	Ad hoc On Demand Distance Vector Protocol
CA	Certification Authority
CC	Coded Cooperation
CF	Compress and Forward
DF	Decode and Forward
DH	Diffie-Hellman algorithm
DoS	Denial of Service
DSA	Digital Signature Algorithm
DSR	Dynamic Source Routing protocol
DTLS	Datagram Transport Layer Security protocol
ECC	Elliptic Curve Cryptography
ECDH	Elliptic curve Diffie–Hellman algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
HI	Host Identifier
HIP	Host Identity Protocol
HIP-BEX	HIP Base EXchange
HIP-DEX	HIP Diet EXchange
HIT	Host Identity Tag
IBAKE	Identity-Based Authenticated Key Exchange
IKE	Internet Key Exchange

CGA	Cryptographically Generated Address
IOT	Internet Of Things
IPsec	Internet Protocol Security
LHIP	Lightweight HIP
M2M	Machine-to-Machine
MAC	Message Authentication Code
MANET	Mobile Ad hoc Networks
MIKEY	Multimedia Internet KEYing
N_QR	New Quality of Recommendation
PMK	Pre-master key
QR	Quality of Recommendation
R_QR	Real Quality of Recommendation
RSA	Rivest Shamir Adleman algorithm
SA	Security Association
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TLS-PSK	TLS- pre-shared key
TMS	Trust Management system
UDP	User Datagram Protocol
WSN	Wireless sensor networks

INTRODUTION

A major trend of today's Internet is its extension into domains, scenarios and even objects that all would have been considered unrelated to Information and Communications Technologies a few decades ago. Energy management, personal health monitoring, safer transportation systems, to name a few frameworks, benefit from the proven design of Internet protocols and become part of a global connected world whose foundations lay in the first packet switched networks and in the TCP/IP protocol suite.

In fact, it was not the Internet protocols themselves that initially opened new domains to interconnection with the legacy Internet architecture. More useful were advances in energy-efficient radio technologies and protocols, which were the essential bricks to design small size autonomous communicating modules, able to monitor and act upon the physical world. First Wireless Sensor Networks (WSNs) relied on leaf nodes that were gathering data about the physical environment and delivered it to a central collecting node, often known as the sink node. This latter could be (and often, was) an IP node, part of the legacy Internet and, as such, remotely accessible and manageable.

Today's transition from legacy WSN systems to the Internet of Things (IoT) can be in a first approach summarized as an extension of the Internet boundaries up to the leaf devices. Instead of stopping at the sink node, as was the case in WSNs, Internet protocols can now run between any two IoT nodes. Accordingly, the architectures and communication types in the IoT are becoming closer to those of legacy Internet. Decentralisation is appearing within once-monolithic, sink-centric sub-systems whose end nodes are now able to be involved in peer-to-peer, bidirectional communications with any remote Internet peer.

Figure 1 schematically depicts the transition of Internet subsets dedicated to the monitoring of physical assets, from Wireless Sensor Networks to the Internet of Things. It highlights the existence of an intermediary step, namely Machine-to-Machine (M2M) communications. The M2M paradigm considers that all nodes can communicate with each other on a peer-to-peer basis, but restricts the application of such communications to a single scenario (e.g. home automation or energy management).

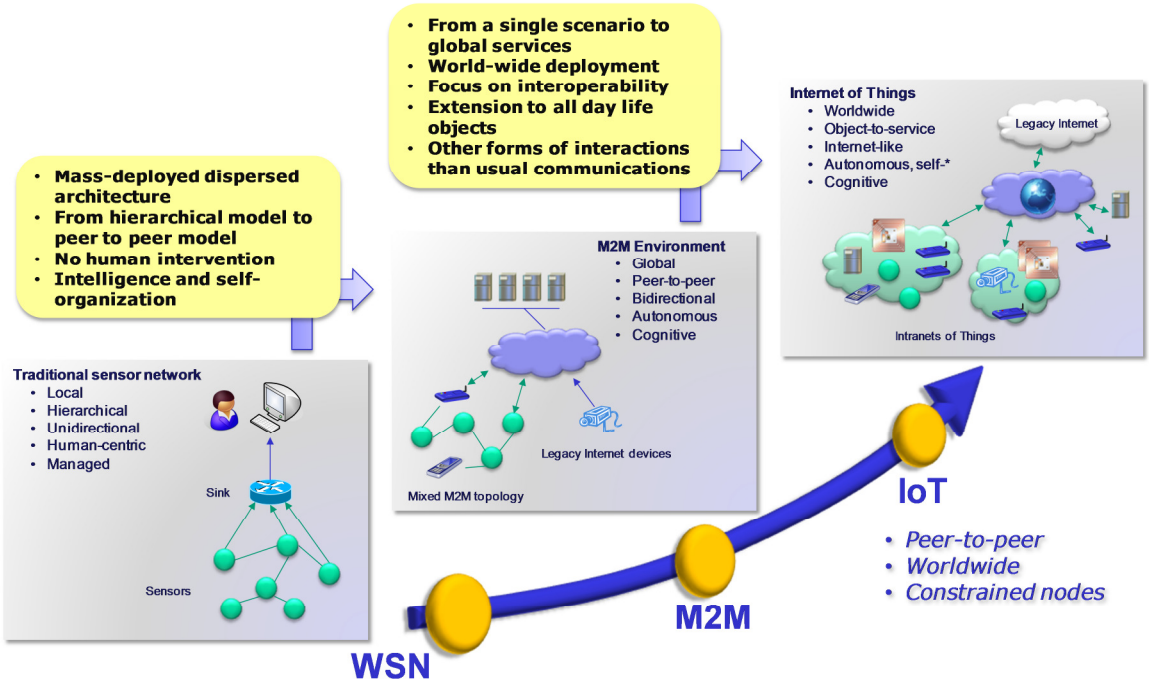


Fig. 1. From Wireless Sensor Networks (WSNs) to the Internet of Things (IoT).

Figure 1 also highlights another characteristic of the transition from WSN to the IoT: the evolution from a human-centric management to autonomous behaviours. This evolution goes along with a

parallel trend in legacy Internet, in which self-* systems (e.g. self-monitoring or self-healing) are emerging. It is even more worthy in unattended, scattered and largely vulnerable (to attackers, radio channel changing conditions or faulty nodes) topologies, such as those considered in the WSN, M2M or IoT architectures. Autonomy can be defined as a local (node) or global (system) ability to monitor the environment, to induce measures needed to correct a foreseen or ongoing incident and to eventually apply the best corrective action. This qualitative description can be mapped to a numeric process, wherein a value obtained as a function of a set of parameters and expressing the overall node or system efficiency, has to be maximized. Among autonomous processes, adaptive ones can be distinguished from cognitive ones. The former merely apply the same function to varying observed parameters, leading to always choosing the same answer if confronted to the same contextual situation. The latter introduce a learning step as part of their reasoning operation, which makes them aware of the results of their last decision. As a consequence, they dynamically update the performance evaluation function used to identify the best action to undertake. The node, or system, will therefore not answer identically to identical situations.

The mere delivery of data from a node to another is the most elementary service in which autonomous processes take place. Basic IP routing is essentially an adaptive process, wherein resilience of a service (packet delivery) can be achieved even though incidents (faulty routing nodes) happen, through a specified monitoring and planning operation (routing table update). Likewise, the ability of networked nodes to exchange information with one another in a dynamically shared radio environment involve adaptive or even cognitive processes that aim at optimizing the use of a scarce resource, namely the radio spectrum. In both cases, autonomy is complemented with collaboration: various nodes collaborate with each other in order to perform end-to-end delivery of an IP packet or to achieve best usage of a radio channel.

The ability for any two nodes of exchanging information with one another is however not sufficient for a networked architecture being deployed in proximity of the physical world (either sensed or acted upon) and therefore vulnerable to malicious attacks on nodes and/or communications channels. Security is another essential service that has to be provided. Here again, autonomy and collaboration offer valuable advantages for the optimisation and resilience of security services. Before going into the details of how autonomous collaborative security services can be profitable in M2M or IoT topologies, it is worth giving a quick overview of how security functions can be categorized in these environments.

Classification of information security functions is often approached with the objective of performing a risk assessment for a system and to eventually develop countermeasures to identified threats. As such, classes of security functions correspond to main families of attacks, wherein an attacker may attempt to alter information (*integrity* security concept), to access sensitive information (*confidentiality* security concept) or to disrupt information-processing services (*availability* security concept). Depending on the scenario, the integrity/confidentiality/availability kernel can be extended to include other security services such as non-repudiation.

Things are somewhat different when considered from the viewpoint of a legitimate member of a protected topology. For example, the security procedures applied to set up integrity protection and confidentiality services between two nodes are very similar: generally, an authenticated key exchange protocol, leveraging on nodes' respective credentials, is invoked; a key derivation/diversification function follows; eventually, the generated keys are used to compute message authentication codes and/or to run symmetric encryption/decryption algorithms. The whole process is reiterated whenever secure (confidential and/or integrity-protected) communications have to be established with a new peer, or when a given key material expires. On the other hand, availability at node's side merely relies on security by design (e.g. use of protocols resilient against Denial of Service attacks) –without requiring active involvement of the node or the use of a dedicated security procedure.

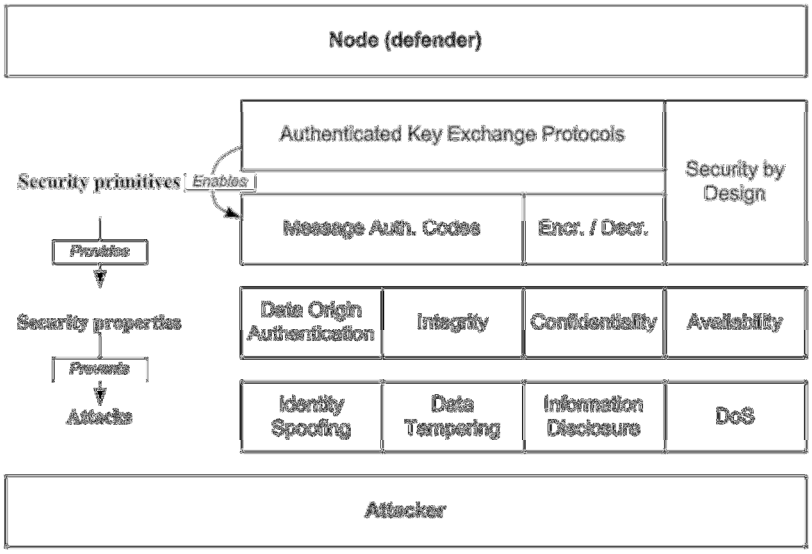


Fig. 2. Schematic view of the main security threats and corresponding countermeasures.

Figure 2 provides a schematic view of how the three main security properties (integrity, confidentiality and availability) relate to their associated security primitives and how they answer the corresponding possible attacks. At defender's side, authenticated key exchange protocols represent the bulk of security primitives used for integrity and confidentiality. However, they rely on computationally heavy cryptographic operations, which may prevent their use by constrained nodes, limited in terms of computing power and/or battery.

This limitation is problematic for a wide range of nodes, found in M2M and IoT scenarios, which precisely exhibit these constraints in both computing power and battery capacity. On one hand, these constrained nodes are involved in end-to-end transactions with remote peers, as required by the decentralized characteristic of the considered scenarios. On the other hand, the prerequisite for any secure channel setup, that is, key establishment, could be either unaffordable or prohibitively expensive for these nodes. While latency could be induced, this is not where the main problem lies: a key establishment operation occurs indeed at the beginning of a novel communication without affecting it afterwards, except when rekeying is needed. For example, a lengthy key establishment phase, in the order of a few seconds or dozens of seconds, would still be acceptable if it occurred only once a day. More critical are the consequences in terms of energy consumption. Battery-powered sensor nodes can be disseminated in hazardous environments. Some are built-in within products and are expected to have at least the same lifetime as their hosts. Changing a discharged battery could therefore be either demanding, or unacceptable. This even without considering the consequences on other neighbouring nodes, which may find themselves disconnected from the infrastructure if their default route passed through a battery-depleted node.

This is where collaboration comes into play. It can be expected, from the heterogeneous aspect of M2M and IoT scenarios, that the architecture containing the constrained nodes also hosts unconstrained ones. We proposed to take advantage of this heterogeneity to involve said unconstrained nodes in a collaborative key establishment process, wherein they would make available to otherwise hindered peers their computing and energy capabilities. By delegating the computationally expensive tasks to a set of peers, a constrained node could thus establish secure, end-to-end communication channels with remote peers instead of relying on inefficient or vulnerable lightweight alternatives that include static shared secrets or use of an intermediary security gateway.

The reliance on collaboration for any kind of service, and even more for the fulfilment of a security service, should however be done on a controlled basis. Collaboration per se may indeed open the way to a new class of attacks, all the more insidious as they would involve internal attackers. Having already passed cryptographic filtering barriers during network access control procedures, these latter have to be identified and excluded based on their behaviours only. This amounts, in a nutshell, to introducing the concept of trustworthiness within a networked architecture. As can be expected, trustworthiness can be difficult to measure when different nodes providing different services have to

be assessed by the same trust management system, especially when these nodes, subject to regular exhaustion of their (low) resource capabilities, become temporarily unable to provide assistance to their peers without being nevertheless to be qualified as malicious. Of course, truly malicious nodes do exist too and have to be dealt with, even though these would likely try to fail the trust metric by camouflaging their misbehaviours.

Like the collaborative key establishment mentioned above, a trust management system is also a security system instantiated on a collaborative basis, wherein multiple nodes share their views about one another's trustworthiness in order to exclude misbehaving nodes from future selections. The present PhD thesis therefore approaches IoT security from two complementary levels that leverage on similar relationships patterns. On one hand, we identified key establishment as the most crucial security procedure in the setup of secure channels, and proposed a novel collaborative key establishment approach for adapting it to highly resource-constrained nodes. On the other hand, we identified trust management as an essential autonomous security procedure for making viable collaborative solutions and proposed a cognitive approach for handling it. Meanwhile, both levels of collaborative security had to be thoroughly tuned in order to take advantage of (when possible), or at least to be resilient against the heterogeneity in nodes, capabilities and services that characterize today's emerging M2M and IoT architectures.

OBJECTIVES AND CHALLENGES

The main objective of this thesis is to design a collaborative solution for end-to-end key establishment in heterogeneous environments. This objective encompasses the following challenges, which are to be specifically addressed:

- Design of a collaborative key establishment system answering the constraints and characteristics of heterogeneous Machine to Machine or Internet of Things environments. For this purpose, these constraints and their impact on the keying design decisions are to be investigated.
- Adaptation to existing key establishment modes and protocols. The designed key establishment protocol will have to leverage on existing key establishment modes (namely key transport, key agreement and key distribution), highly different to one another, and for which collaborative embodiments will have to be designed –if these modes are judged suitable for the Internet of Things. Likewise, the proposed collaborative solution will have to fit within the scope of current key establishment protocols (similar syntax and authentication model).
- Security of the proposed collaborative scheme against malicious players. Relying on a collaborative process, the developed key establishment solution will indeed be exposed to attack schemes targeting its early design. In order not to be self-contradictory, the security system we design must be resilient against these attacks. Security by design and autonomous security will be the key to protect it against information disclosure and Denial of Service attacks. Special care will be taken to protect the established key as well as to exclude from the collaborative process the malicious or faulty nodes.
- Evaluation of the proposed key establishment solution. In order to be satisfactory, the developed key establishment protocol and its accompanying security framework must be validated both in terms of security (formal security analysis whenever possible, rigorous simulation of attacks otherwise) and performance (usability by constrained devices).

CONTRIBUTIONS

In order to reach the planned objectives, the following contributions were produced.

- An overall overview of key establishment schemes and protocols was carried out. Its results were confronted to a study of Internet of Things characteristics and requirements. Accordingly, relevant key establishment protocols, belonging to the key transport and key

agreement families were identified. A study of how to securely and efficiently design collaborative versions of these protocols was conducted. This work is to be published in:

  - Y. Ben Saied, A. Olivereau, M. Laurent and D. Zeghlache, *Lightweight collaborative keying for the Internet of Things*, submitted to Elsevier Ad hoc Networks, 2013.

- The technical design of collaborative key establishment schemes led to the development of two classes of solutions, respectively adapted to the key transport and key agreement families. Complementarily, we designed a framework for lightweight authorisation of assistant nodes (lightweight signing and validating). We also focused on the development of performance evaluation techniques: the security of the developed solutions was formally proven using the AVISPA tool. We also designed a quantitative performance evaluation model which allowed us to compare the energy cost of the developed solutions to other key establishment protocols, with respect to both computations and data transmissions. Finally, we developed resilience schemes allowing collaborative keying to withstand faulty assisting nodes. These contributions were published in:

  - Y. Ben Saied, A. Olivereau and D. Zeghlache, *Energy Efficiency in M2M Networks: A Cooperative Key Establishment System*, 3rd International Congress on Ultra-Modern Telecommunications and Control Systems (ICUMT) 2011.
  - Y. Ben Saied, A. Olivereau and D. Zeghlache, *Etablissement de clé de session en environnement M2M entre nœuds à ressources fortement hétérogènes*, Computer & Electronics Security Applications Rendez-vous (C&ESAR) 2011.
  - Y. Ben Saied and A. Olivereau, *HIP Tiny Exchange (TEX): A Distributed Key Exchange Scheme for HIP-based Internet of Things*, 3rd International Conference on Communications and Networking (ComNet) 2012.
  - Y. Ben Saied, A. Olivereau and M. Laurent, *A Distributed Approach for Secure M2M Communications*, 5th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2012.
  - Y. Ben Saied and A. Olivereau, *D-HIP: A Distributed Key Exchange Scheme for HIP-based Internet of Things*, First IEEE WoWMoM Workshop on the Internet of Things: Smart Objects and Services (IoT-SoS) 2012.
  - Y. Ben Saied and A. Olivereau, *(k, n) Threshold Distributed Key Exchange for HIP based Internet of Things*, 10th ACM International Symposium on Mobility Management and Wireless Access (MOBIWAC) 2012.

- The need to reinforce our collaborative approaches with a solid trust model was highlighted in our previous studies. We therefore conducted a survey on collaborative systems security management. The synthesis of this survey led us to identify a set of best practices for the design of a Trust Management System in the framework of Internet of Things. This work is to be published in:

  - Y. Ben Saied, A. Olivereau, D. Zeghlache and M. Laurent, *A Survey of Collaborative Services in Modern Wireless Communications and their Security-related Issues*, submitted to Elsevier Journal of Network and Computer Applications, 2013.

- In accordance with the identified best practices, we specified a novel trust management system, named COACH (COntext Aware and multi-service trust model for Cooperation management in Heteregenous wireless networks) and highlighted how our proposed trust management system can be compared with the state of the art solutions proposed for enabling various collaborative networking services. These contributions were published in:

  - Y. Ben Saied, A. Olivereau and R. Azzabi, *COACH: a COntext Aware and multi-service trust model for Cooperation management in Heteregenous wireless networks*, 9th International Wireless Communications and Mobile Computing Conference (IWCMC) 2013.

STRUCTURE

This thesis report is organized as follows. We start in chapter 1 with a review of the challenges introduced by the transition from legacy Internet to Internet of Things, especially from the viewpoint of security. We highlight the relevance of the key establishment problem and its general inconsistency with nodes constraints. We assess the adaptability to the IoT paradigm of the legacy Internet protocols, as well as that of ad-hoc solutions purposely designed to fit the needs of constrained devices; we conclude on the inadequacy of either to manage end to end security associations involving highly constrained nodes.

Consequently, we introduce the concept of collaborative key establishment in chapter 2, with the objective of providing a means for highly constrained nodes to establish end-to-end secured contexts with distant peers. New collaborative key establishment techniques are proposed for key transport and key agreement schemes. Accordingly, we details the prerequisites and bootstrapping approaches for these techniques, as well as their actual embodiments within the retained security protocols identified in previous chapter. We also provide a detailed performance evaluation from the points of view of security (formal security analysis) and energy consumption (evaluation of computation and communication energy costs) that proves the pertinence of our proposed key establishment approach.

With collaborative key establishment arises the need to choose the best peers to outsource security functions to, a recurring need in collaborative processes. For this reason, we explore in chapter 3 the collaborative solutions for networking services that exist in the literature, as well as the security mechanisms that are designed to protect them. From this study, we identify both useful design choices and improvable areas, especially with respect to applicability of a common trust management system to a wide range of collaborative services, involving nodes whose resource availability is expected to vary a lot over time.

These considerations lead us to propose in chapter 4 a novel trust management system for collaborative networking services in the Internet of Things. Along with the specification of this system, we pay a particular attention to its behaviour when subject to a class of attacks specifically designed to target trust management systems. The results show that our proposed system is able to withstand these attacks more efficiently than its counterparts that exist in the literature.

Chapter 1: KEY ESTABLISHMENT IN THE INTERNET OF THINGS

1.1. INTRODUCTION

The heterogeneous nature of Internet of Things (IoT) architecture, involving a wide variety of entities with different resource capabilities, makes it challenging to provide end-to-end secured connections. A direct use of existing key exchange schemes between two IoT entities may be unfeasible unless both entities be able to run the (expensive) cryptographic primitives required to bootstrap them – thus leaving aside a whole class of resource-constrained devices. Clarifying how existing security protocols can be adapted to fulfil these new challenges still has to be improved. In this chapter, we revisit existing end-to-end security standards and key establishment schemes and discuss their limitations considering the specific scenarios of the IoT. After having defined in section 1.2 the concepts that underlie the Internet of Things, we introduce in section 1.3 the technical elements that will help us to characterize a key establishment protocol. We then carry out an in-depth study of the key establishment solutions that have been proposed for constrained devices, from legacy WSNs to Internet-integrated pre-IoT topologies. We conclude this chapter in section 1.5.

1.2. FROM LEGACY INTERNET TO THE IOT

The current transition from legacy Internet to Internet of Things (IoT) involves multiple changes in its communication paradigms. The diversity of scenarios where internetworked entities have to exchange information with one another without human interaction is increasing and is planned to extend to almost all environments, from individual customers' everyday life to industrial processes. Accordingly, more and more objects become able to communicate, following as a rule of thumb an always greater interaction with the physical world, which is not only timely and accurately sensed but also understood and acted upon. Wireless sensor networks [1] were the first step in this direction.

Wireless sensor networks (WSNs) consist of a large number of physical devices, geographically close to one another, deployed inside a monitored site, and communicating together in a wireless multi-hop manner. These devices target the same objective: the wireless infrastructure they build aims to detect events that take place in the monitored environment and convey results of sensing to a small number of dedicated gateways called sinks, which eventually send aggregated data to remote management units. WSNs are widely applied in a large number of monitoring applications such as military, environmental, health, home and industrial applications. Usually sensor nodes are small and inexpensive devices powered using batteries, so that their capabilities in terms of both energy and computing resources are highly constrained. Accordingly, optimizing energy consumption has been the key motivation in the research field of sensor networks. Proposed protocols and applications are being designed keeping in mind energy efficiency. More recently, energy harvesting technologies have been proposed for the same goal of maximizing the lifetime of the WSNs: a sensor node can be able to draw energy from the environment and supplements its battery, as long as it disposes of a harvesting circuit and a nearby convertible energy source such as light, wind or vibrations.

Machine to machine (M2M) environment, largely extending the sensor networking model, represents a more advanced type of network referring to data communication between physical devices without human intervention [2]. M2M networks inherit resource-limited, un-guarded and mass deployed nature of sensor networks while developing it through embedded intelligence and self-organisation. The Machine to Machine (M2M) paradigm can be characterized by three main features. First, it involves a highly diversified pool of components, ranging from low-resource sensors to

powerful servers, these components being distributed over a large geographical environment. Second, it emphasizes the increase of autonomy, as compared with legacy Internet. While all of the M2M systems are designed to provide decentralisation and minimize the requirement of human involvement, most advanced ones may even implement functions of situation awareness, self-organisation or cognition. Finally, M2M systems adopt a distributed communication model wherein any two nodes may establish relationship with each other, provided that one is offering the service, or resource, which is needed at the other end. To that respect, M2M systems broke the logical and topological simplicity of sensor networks. Contrary to what happens in WSNs, the communication path between two nodes does not have to follow a hierarchical path, e.g. from sensor to sink, and from sink to remote management units. A sensor in an M2M environment will likely have direct communications with other peers irrespective of their distance, role and capabilities, provided that these relationships are desirable from the viewpoint of the M2M scenario. This novel paradigm, wherein nodes communicate with a large set of heterogeneous entities through a decentralized pattern, leads to situations where unbalanced resource capabilities between the two communicating peers are confronted.

The Internet of Things further extends the M2M paradigm into two directions. First, it aims to interconnect much wider sets of objects, even those that were not natively supposed to be able to communicate. Barcodes and tags allow otherwise inert objects to advertise their presence and sometimes to receive and store information. This makes them part of the connected world. Second, the IoT targets universality and global interoperability whereas most M2M architectures are dedicated to the fulfillment of a given task, be it wide-scale (e.g. Smart Grid operation [3]) or small-scale (e.g. home automation [4]). The advantages of interconnecting huge sets of “things” belong to the fields of adaptation (ability to sense / act on the environment) and autonomous orchestration of new services (interactions appear when entities discover each other, along with their needs and capabilities). In this perspective, IoT is defined as a global architecture featuring a large number of heterogeneous players with a wide variety of mechanisms and scenarios, hence leading to the vision of “anytime, anywhere, any media, anything” communications.

1.3. REVIEW OF KEY ESTABLISHMENT SCHEMES

Like legacy Internet nodes, IoT nodes require security for their communications. The major requirements related to security concern authentication, confidentiality, non-repudiation and data integrity. These security services rely on the use of cryptographic primitives consisting of encryption/decryption and signature/verification schemes. In turn, these primitives require an initial key establishment process that must fit to the low capabilities and cost constraints of IoT components, most of which cannot implement complex security schemes. Key establishment protocols exist in today's Internet. However, the underlying cryptographic algorithms are either too heavy to run on resource-constrained nodes, or do not provide a satisfactory security level.

Key establishment protocols, also named key exchange protocols, are used to "*provide shared secrets between two or more parties, typically for subsequent use as symmetric keys for a variety of cryptographic purposes*" [5]. These purposes include the use of symmetric ciphers and message authentication codes, which are in turn used as security primitives for enabling various security protocols such as source authentication, integrity protection or confidentiality.

The word "protocol" in the above definition could be misleading, because it is used in multiple contexts in which its sense changes slightly. It has thus to be clarified first.

1.3.1. Algorithmic protocols, communication protocols

A protocol can be defined as "*a multi-party algorithm, defined by a sequence of steps precisely specifying the actions required of two or more parties in order to achieve a specified objective*" [5]. This definition however encompasses two kinds of protocols that exist in the world of telecommunications and that collide in the field of security. On one hand, classical communication protocols of the OSI model – as specified for example in the Internet Engineering Task Force – define

how two or more networked entities interoperate. These protocols include precise packet format specification along with state machine definitions. On the other hand, cryptographic algorithmic protocols define how two or more logical entities carry out a cryptographic operation. They define mandatory elements for doing so, such as the data structures that have to be transported, and the corresponding order. They do not specify, however, how data are to be transported (e.g. encoding, optional parameters, resilience support, networking parameters…).

Let us take the example of the key establishment operation for the IPsec protocol[1]. The key establishment *communication* protocol for IPsec, in the sense of the first definition, is the Internet Key Exchange (IKE, [12]) protocol. However, the key establishment *algorithmic* protocol for IPsec, in the sense of the second definition, is the cryptographic protocol on which IKE relies, that is, the Diffie-Hellman protocol. This distinction is clear and easily understandable. Things become more complex however when a single communication protocol, such as EAP, can leverage on a multitude of distinct algorithmic protocols. Complexity increases even more when the algorithmic protocol within a well-known telecommunication security protocol such as TLS can be entirely modified through the mere change of one bit in the handshake sequence. The distinction between these types of protocols is therefore of high importance. Unless otherwise stated, this chapter deals with cryptographic algorithmic protocols[2].

1.3.2. Classification of key establishment protocols

Key establishment protocols can be classified according to three criteria: the key delivery scheme (key transport or key agreement), the underlying cryptographic primitive family (symmetric or asymmetric) and the authentication method. The number of involved peers[3] (two, peer-to-peer or three, server-assisted) is sometimes added to these criteria. These notions are discussed in what follows.

1.3.2.1. Key transport vs. key agreement

A two-party *key transport* protocol is a protocol that runs between two peers, in which one or more secret value(s) are generated at one or both peers and securely transferred to the other peer. The resulting key is obtained as a function of the transferred secret values and possibly of other parameters that may have been exchanged as part of key transport.

In a one-pass key exchange, only one secret value is sent from one of the peers to the other. The established key may be either this secret value itself, or may be derived from it along with other parameters, such as nonces. In a two-pass key exchange, both peers exchange secret values that are used as input for the key generation function. Note that it is generally not safe to let one partner entirely control the key value.

A variety of server-assisted key transport is the distribution of a session key from a central server (key distribution center) to two peers. This requires, of course, that the central server be able to perform the distribution in a secure manner, e.g. through pre-established secured channels to both peers. Another, less frequent, variety of server-assisted key transport consists for the server to let one peer generate the session key, obtain it from this peer, and retransmit it over another secure tunnel to the second peer. In this second variety the assisting server is called a key translation center.

A two-party *key agreement* protocol is a protocol that runs between two peers, in which the resulting key is derived at both peers from public information exchanged between the peers. While said public information might take the form of an encrypted secret, the decrypting of this encrypted secret by either the recipient peer or by the originating peer itself is never required.

The Diffie-Hellman (DH) protocol [6] is the best known and most widely used key agreement protocol. It requires that two peers A and B first agree on appropriate prime (p) and generator (g). Then, A and B choose secret values, respectively a and b, compute the corresponding public values, respectively $g^a \bmod p$ and $g^b \bmod p$, and exchange these public values with each other. The same

[1] Actually, a protocol suite made of the AH and ESP protocols.

[2] In the literature, these protocols can also be designated as "methods", "algorithms" or "sub-protocols".

[3] A key establishment protocol runs between two or more parties. In this thesis, we focus on peer-to-peer (pairwise) key establishment and do not consider the joint setup of a group key between more than two parties.

Diffie-Hellman shared secret K is then obtained at A by computing $(g^b \bmod p)^a$ and at B by computing $(g^a \bmod p)^b$. The protocol exchange is depicted in figure 3 below:

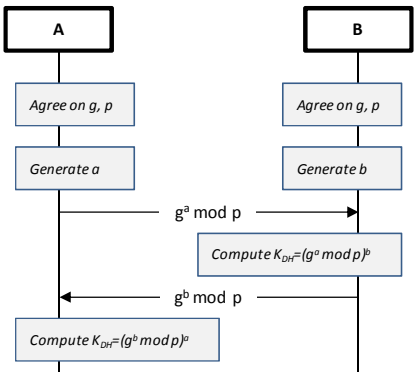


Fig. 3. Diffie-Hellman key agreement.

An often claimed security property of the Diffie-Hellman protocol is the perfect forward secrecy. This property ensures that the established secret could not be retrieved even though all long-term secrets of both peers are divulged. In the base Diffie-Hellman protocol, *a* and *b* are random numbers that are dynamically chosen as part of the key management protocol and immediately erased from memory afterwards. They could therefore not be qualified as "long-term secrets", which ensures that the Diffie-Hellman protocol fulfils the perfect forward secrecy property. This should not be generalized to all key agreement protocols, though. Some key agreement protocols are based on key pre-distribution. For example, the variant of the Diffie-Hellman protocol used in the HIP-DEX key establishment communication protocol (reviewed in what follows) requires that the Diffie-Hellman secrets *a* and *b* be statically fixed and remain the same in all key establishment operations. This use of Diffie-Hellman leads to losing the perfect forward secrecy property that is generally associated with it.

1.3.2.2. Cryptographic primitives

Both key transport and key agreement exist in embodiments that rely either on symmetric or on asymmetric cryptography. These cryptographic primitives should not be confused with those of the authentication mechanisms that may be integrated with the key establishment protocol and that are the subject of the next classification criterion. To clarify this distinction, let us take again the example of the Diffie-Hellman key agreement protocol. Diffie-Hellman is based on asymmetric cryptography primitives (actually, most of the key agreement protocols are). Yet Diffie-Hellman, natively unauthenticated and vulnerable to man-in-the-middle attacks, has to rely on authentication techniques, some of which can be based on symmetric techniques.

Considering only the key delivery scheme and the cryptographic primitive type, four cases are possible:

- Key transport based on symmetric cryptographic primitives. This category regroups algorithms in which two peers, already owning a shared key, derive another one. Such operation typically happens when a symmetric key has to be refreshed, or when an ephemeral secret (e.g. transient session key) has to be derived from a long-term one.
- Key transport based on asymmetric cryptographic primitives. In this category are found various key establishment protocols ranging from simple one-pass encryption of a secret key with a public key to more complex X.509 keying protocols.
- Key agreement based on symmetric cryptographic primitives. A corresponding protocol, Blom's scheme, is presented in [1]. Although interestingly dissociating the key agreement notion from the Diffie-Hellman protocol, one cannot but notice that such algorithmic protocols are not used by main (and even minor) communication protocols.
- Key agreement based on asymmetric cryptographic primitives. With rare exceptions, this category is composed of the Diffie-Hellman protocol and its variants.

1.3.2.3. Authentication method

Authentication for a pairwise key establishment protocol relates to the ability, for one or both nodes that undertake it, to bind the established key material with the identity of its peer. While it is generally a good thing to have a pairwise key establishment protocol authenticate both peers to each other, it is not always the case. Commonly, only one peer is authenticated to the other; the authentication of the other peer, if required, has then to be ensured by another mechanism, possibly at another layer.

Authentication brings us back to the distinction we introduced in the beginning of this chapter between algorithmic and communication protocols. Some algorithmic protocols natively provide authentication. This is the case, for example, of a one-pass key transport protocol wherein a session key k is sent from a node A to its peer B, encrypted with B's public key. This protocol achieves indeed more than confidential key delivery: it proves to A that a node knowing k must be identified as B, since only B is expected to have been able to decipher the message containing k[4]. On the other hand, as mentioned above, the Diffie-Hellman protocol does not natively provide authentication. The Diffie-Hellman public values have therefore to be authenticated at communication protocol level, as is done by the IKE protocol, which ensures through digital signatures or keyed hashes that their origins can be validated.

Like those of key establishment protocol, the cryptographic primitives that underlie the authentication method can be classified as symmetric vs. asymmetric techniques. With the objective of defining the best practices for an IoT key establishment protocol, it is worth, though, going beyond this distinction and considering the underlying identity models. The categories of authentication that can be distinguished are listed hereafter. For clarity reasons, this list is made simpler by assuming that mutual authentication is desired, and that both peers use the same authentication method to each other.

- Shared secret –based authentication. This is the classical symmetric authentication scheme wherein two parties are statically configured with, or otherwise acquire, a common shared secret mapped to their respective identities.
- Static public key authentication. In this asymmetric authentication scheme, the two parties are statically configured with their respective public keys, mapped to their respective identities. Proving the knowledge of the corresponding private key implicitly ensures ownership of the matching identity.
- Certificate-based authentication. This is a variant of the previous category, wherein the mapping of a public key to an identifier is not a static configuration parameter but is obtained in the form of a signed certificate. Certificate-based authentication requires that a third party, the certificate authority, be trusted by both authenticating peers.
- Cryptographically generated identifiers. This family of asymmetric techniques changes the implicit assumption that any kind of identifier can be authenticated, provided that it is securely bound to a public key. These techniques assume indeed that the authenticated identifier of a node is obtained from the node public key, e.g. in the form of a hash of this public key. Mechanisms are then defined in order to build protocol stack identifiers (typically, IPv6 addresses) from these cryptographically generated identifiers.
- Identity-based authentication. This last set of asymmetric techniques bases on the Identity Based Cryptography paradigm wherein, oppositely to the previous category, a node's public key is derived from its identity (whatever the format of this identity). Like in all asymmetric techniques, a node proves its identity by providing a proof of knowledge of the corresponding private key.

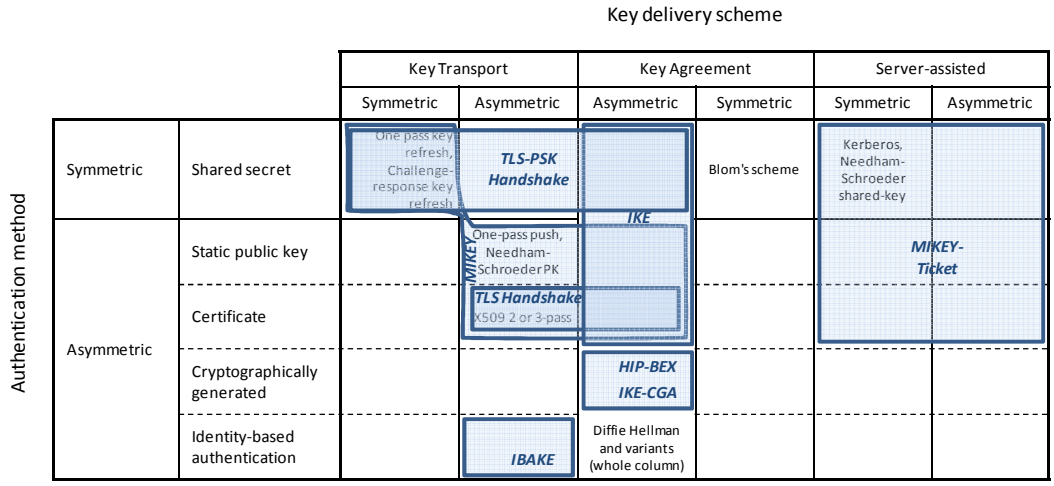
1.3.2.4. Synthesis

Our objective is here to provide a global view of the existing algorithmic protocols, in order to ease the identification among them of the best candidates for IoT key establishment. This synthetic global view is provided in the form of a table, on which we chose to superpose the most known/used *communication* protocols. Usability of *algorithmic* protocols within communication protocols currently in use in today's Internet is indeed a criteria that should not be left apart: the Internet of

[4] The two steps of ensuring that only B may know the key k and obtaining the proof that some node knows the key k are respectively designated in [5] as *implicit key authentication* and *key confirmation*. Together, they form the *explicit key authentication* property.

Things will definitely not start with a "clean slate" design approach, but will likely have to interoperate with widely adopted protocols of legacy Internet.

Table 1. Classification of key establishment protocols according to the key delivery scheme and authentication method, with main key establishment communication protocols represented in overlay.



As can be seen in Table 1, the existing key establishment communication protocols mainly base on asymmetric cryptography, be it for the delivery/agreement scheme itself, or for the authentication method implemented within the protocol. Empty cells in the table are mostly found in the symmetric key transport and symmetric key agreement columns. Symmetric key transport protocols do exist, though; however, they mainly consist in key refresh / key derivation protocols, which we found did not fully qualify as key establishment protocols. Only the MIKEY [7] and TLS-PSK [11] protocols are included in the column, since they are used to distribute session keys from long-term shared keys. Symmetric key agreement protocols are uncommon and require complex setup (pre-distribution).

1.3.3. IoT key establishment: generic design decisions

This subsection reviews the generic design decisions that are involved in the identification of a key establishment protocol for the Internet of Things. These decisions fall into four main categories: those that are related to the fulfilment of security requirements, those that are related to pervasiveness (the Internet of Things is to encompass a wide variety of devices and networks, including legacy Internet), those that are related to efficiency (among IoT devices, some are resource-constrained) and those that are related to adoptability or interoperability (the IoT should preferably use proven and deployed technologies and protocols).

1.3.3.1. Security

Contrary to wireless sensor network security, security in the IoT context involves end-to-end communications. The decentralized and bidirectional IoT communication paradigm also rules out the definition of static *client* and *server* roles: depending on the context, it is expectable that an IoT node will act alternatively as a client and as a server. These considerations translate into two security requirements. On one hand, end-to-end security should be provided. This means that only the two participants involved in the pairwise key exchange protocol should have access to the generated key. On the other hand, mutual authentication has to be provided. The two peers that establish a key between them should in the meantime authenticate to each other and bind the generated key to their respective identities.

1.3.3.2. Pervasiveness

By qualifying the Internet of Things as "pervasive", we refer to its foreseen universality, as a communication network interconnecting much more nodes than today's Internet, and actually

encompassing this today's Internet. Pervasiveness puts additional requirements on a key establishment protocol for the IoT. Especially, it makes it highly unlikely that two nodes wishing to generate a key between them can leverage on a pre-existing security relationship based on long-term shared secrets or static public keys. For this reason, dynamic asymmetric key delivery schemes and authentication methods should be favoured when designing an IoT key establishment protocol.

Pervasiveness also means that any two nodes may have to interoperate with each other, without considering their respective nature. Special care should therefore be taken, when designing an IoT key establishment protocol, to make sure that two nodes with important differences in capabilities are nevertheless able to generate a key with each other.

1.3.3.3. Adoptability

The Internet of Things will not emerge through the definition of entirely novel protocols. The approaches that rely on key generation schemes or authentication methods of limited usage should not be favoured. Of course, interoperability mechanisms with these latter should be developed when desirable, though.

At this stage, it is worth quickly describing the two most widely adopted end-to-end security protocols we refer to in Table 1.

The Internet Protocol security (IPsec) [8] resides at the Network Layer of the OSI Model, which enables it to function independently of any application. It creates a secure (encrypted and/or integrity-protected) tunnel between two endpoints, through which data can be exchanged safely, without being vulnerable to eavesdropping, packet forging/replaying or sender spoofing attacks.

Like IPsec and unlike hop-by-hop solutions, the Transport Layer Security TLS [9] provides the same end-to-end security services at the transport layer while still being application-independent. Hence it can encapsulate higher-level protocols layering on top of the transport layer protocols. TLS has been designed to work with reliable transport protocols providing in-sequence delivery, such as the Transmission Control Protocol (TCP). Recently, a datagram-oriented variant DTLS [10] has been proposed to operate on top of datagram-oriented transport protocols, such as the User Datagram Protocol (UDP). Both IPsec and TLS have the same design and provide equivalent security measures.

IPsec and TLS security protocols rely on the use of cryptographic mechanisms such as encryption/decryption block ciphers and hash functions, in order to ensure the required security services for a communication. In turn, each of these mechanisms requires an initial key establishment phase allowing two communicating entities to authenticate each other and set up the required cryptographic keys. TLS protocol is preceded by a handshake protocol called TLS Handshake, which is responsible for key establishment and authentication. Likewise, the Internet Key Exchange [12] protocol and the Host Identity Protocol Base Exchange (HIP BEX) [13] are both designed to perform keying for IPsec protocol. In practice, IKE is by far the most widely used IPsec keying protocol. Nevertheless, HIP BEX, the base key exchange mechanism of the Host Identity Protocol (HIP) [13], is gaining momentum in the Internet of Things. Indeed, HIP is a secure protocol that provides not only identifier ownership and identifier/locator split[5], but also supports mobility and interoperability [14], in addition to being based on a mature, proven design, for which various embodiments on different OSI layers have been proposed [15], [16].

Each of these key exchange schemes independently implements specific techniques and cryptographic algorithms to derive a secret key and ensure the required mutual authentication between the endpoints of a communication.

1.3.3.4. Efficiency

Efficiency has always to be considered when designing a new protocol. Four criteria are especially relevant when assessing cryptographic protocol efficiency: the number of exchanged messages, the needed bandwidth, the complexity of computations, and the possibility of pre-computations. Importance of these criteria increases when designing a protocol that will have to be run by highly resource-constrained nodes with low computational power, low memory, and limited battery capacity.

[5] Beyond being reasonable from an implementation point of view, the distinction between *to whom* a data unit should be sent, and *to which location* it has to be routed offers interesting opportunities in the field of IoT, especially for aggregation or resilience purposes, where the identifier/locator bindings can become quite loose.

Overall energy consumption, induced by both computations and message exchanges, is a good metric for these nodes. A protocol will be defined as more efficient than another if it obtains a metric value inferior to that of the other, while providing the same security level. Efficiency requirement is an important concern in the IoT, since we consider that most of its components are resource-constrained and heavy protocols may hinder their integration.

1.3.3.5. Synthesis

From the design decisions reviewed above, we can adapt our initial classification of key establishment protocols in order to identify among them the most suitable to the Internet of Things. The results of this identification are presented in Table 2.

Table 2. Refinement of the key establishment protocols classification. Some candidates are ruled out either because they are not judged secure enough (no end-to-end security), or because they would not meet the IoT pervasiveness requirement, or because their adoptability is evaluated as low with respect to their use as of today. Efficiency is not discussed here, but will be the most important evaluation metric in the next section.

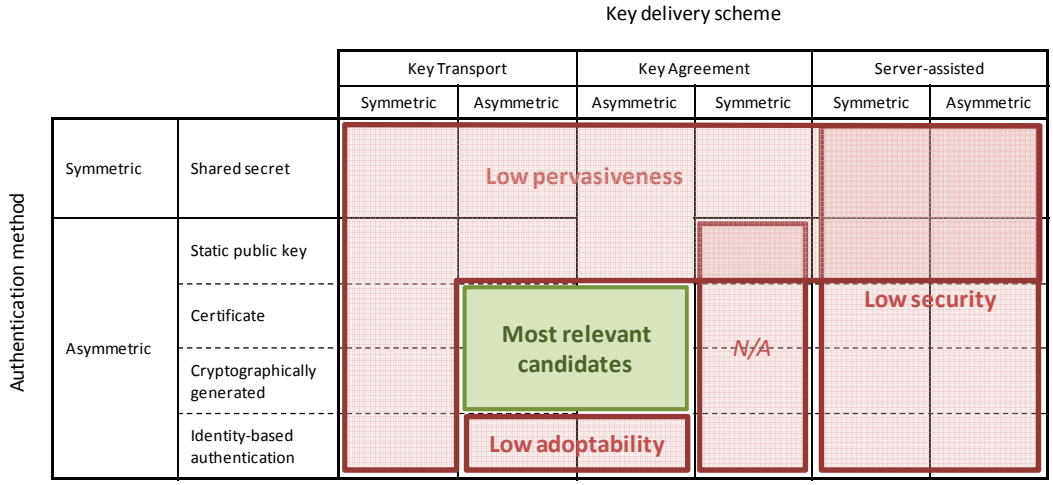


Table 2 was obtained as follows. First, solutions relying on key pre-distribution were discarded, as they did not meet end-to-end security requirements. Then, solutions relying on symmetric cryptography or assuming initial knowledge of peer public key were discarded as they did not meet the pervasiveness requirement. It has to be noted that these first two requirements do not contradict each other: dynamic obtaining of asymmetric public keys through certificate and induced reliance on a certificate authority are different from letting the trusted third party generate the keys for both peers, and be thus in position to launch a key escrow attack. Finally, we discarded the solutions that were based on identity-based cryptography, which we considered not adopted enough.

The most relevant communication key establishment protocol candidates, retained from the juxtaposition of Table 1and Table 2, are summarized in Table 3.

Table 3. Retained key establishment protocols for the IoT before considering the efficiency metric.

Properties / Protocol	Security protocol	Algorithmic protocol	Key delivery scheme	Authentication method
TLS Handshake	TLS	One-pass push / Diffie-Hellman	One-pass key transport / Key agreement	Certificates
Internet Key Exchange (IKE)	IPsec	Diffie-Hellman	Key agreement	Certificates[6]
HIP Base Exchange (BEX)				Cryptographically generated identifiers

1.4. APPLICABILITY OF EXISTING KEY EXCHANGE SCHEMES FOR IOT SCENARIOS AND RELATED WORK

In this section, we assess the retained key establishment schemes of Table 3 from the point of view of the efficiency requirement. These schemes involve heavy asymmetric cryptographic primitives that impact both energy and storage resources of a communicating entity. The resource constraints of most IoT components limit the implementation of these complex cryptographic mechanisms required to perform the key establishment, which could rapidly drain their resources and reduce the network performance. Existing end-to-end security protocols such as TLS and IPsec, with their actual resource intensive key exchange design, could not directly cope with the envisioned scenarios and requirements in the IoT. The feasibility of these security standards has to be revisited to adapt them to the IoT scenarios.

1.4.1. Energy model of a constrained IoT node

We consider the following example system to underline the need to address energy efficiency issues in key establishment protocols: we determine the energy model of the popular sensor node TelosB [17] featuring the 16-bit MSP430 microcontroller with a clock frequency of 4 MHz and operating at a transmission data rate of 250 kbps. TelosB, powered by two AA batteries, runs TinyOS and embeds an IEEE 802.15.4 compliant RF transceiver. This platform is the successor to the Mica family of motes (Mica2dot, Mica2 and MicaZ). It offers lower power consumption and longer battery life compared with the Mica2dot and Mica2.

[6] Shared secret or static public keys IKE authentication methods are not considered here, since they do not meet the pervasiveness requirement. EAP-based IKEv2 authentication would likely rely on a certificate-based EAP method.

Table 4. Energy costs of communication and computational operations on the TelosB platform.

Operating mode	Energy cost
Transmission (1 bit)	0.72 μJ
Reception (1 bit)	0.81 μJ
Symmetric encryption AES-128 (one 128-bit block)	2.47 μJ
RSA-1024 Sign	24.5 mJ
RSA-1024 Verify	1.24 mJ
Diffie-Hellman-1024 public value generation	60 mJ
Diffie-Hellman-1024 shared key generation	105 mJ

Table 4 presents the energy consumption for common communication and computational operations that we obtained for a TelosB platform[7]. Results show that when asymmetric cryptographic operations are performed during a key establishment protocol, the node is observed to consume energy in the order of mJoules. The most demanding operations in terms of energy are the computation of the Diffie-Hellman shared key, immediately followed by the generation of the Diffie-Hellman public values. Signatures computations are also non negligible operations from the point of view of a highly resource-constrained node. Verifications, though, are more affordable since they are 20 times less expensive than their signing counterparts. Being of the order of μJ, the communication (transmission and reception) and symmetric encryption costs are considerably lesser than those of asymmetric cryptography operations.

Putting these cryptographic mechanisms in perspective with each other, as well as in perspective with the overall battery capacity of the TelosB (2x AA-sized NiMH batteries, that is 2x 7.7 kJ) emphasizes how full reliance on heavy asymmetric cryptographic operations to set up shared secrets would speed up the battery drainage. This motivates us to investigate techniques to facilitate energy-efficient execution of security protocols in the context of the Internet of Things.

1.4.2. Related work: energy-efficient key establishment solutions

The need for energy efficient solutions was initially identified to accommodate the resource constraints of WSN nodes. To that respect, the design of a lightweight key establishment system for WSNs was recognized as a highly relevant challenge, and led to the development of several mechanisms. Early on, these mechanisms were adapted to the existing WSN topologies. Since these latter were both highly hierarchical and either disconnected from the Internet, or connected to it by means of dedicated gateways, these systems favoured hop-by-hop security. Another reason for relying on hop-by-hop security was the implementation of security mechanisms at the link layer, which certainly allowed for lightweight communication stacks and more efficient bandwidth management but also restrained the scope of the provided security services (integrity and confidentiality) to one single hop.

Recently, under the umbrella of IoT, integrating sensor nodes with the Internet to support direct communications between Internet hosts and sensor nodes became a challenging goal. Accordingly, sensors communications stacks became more comprehensive and the all-IP paradigm began to look like a viable solution for sensor nodes too. With WSN IP nodes arose the need for end-to-end secure connections between these and remote IP Internet nodes, bypassing the dedicated WSN gateways. The adaptation of the legacy Internet end-to-end security protocols, namely IPsec and TLS, to the constrained WSN systems led to solving a variety of questions. These protocols were initially designed for unconstrained nodes, and their applicability to WSN nodes required to reshape them in terms of state machine complexity, data structures and, mostly, cryptographic primitives. Accordingly, recent scientific works have been proposed that describe lightweight key establishment schemes to efficiently implement IPsec and TLS on constrained devices.

[7] The analytical process for doing so is described in the next chapter.

This subsection reviews the approaches proposed by the international research community with respect to lightweight key establishment for sensor nodes. The reviewed schemes range from the initial solutions proposed for traditional WSNs to the latest approaches that try adapting legacy Internet security protocols to IP-based WSN nodes, considered as being part of a global Internet of Things.

1.4.2.1. Efficient key establishment schemes in traditional WSNs

Several key establishment schemes have been proposed in traditional WSN deployments in order to cope with the resource constraint nature of sensor devices. Most of the proposed approaches rely on symmetric cryptography primitives due to their low resource consumption. Such solutions are considered more efficient for sensor nodes. The most relevant researches are described in what follows.

The simplest solution is to set the same master secret key in all the nodes. Any pair of nodes can then use this global secret key to achieve key establishment and exchange a secret pairwise key. This solution is however highly vulnerable to node compromise, since a successful attack on one node, allowing to retrieve the master secret key, means that the overall network security system is broken. Another key pre-distribution scheme is to let each sensor node carry $N-1$ secret pairwise keys, each pair being shared between this sensor and one of the other $N-1$ sensors (N being the total number of sensors). However, this scheme is unfeasible for sensors with an extremely limited capacity of storage, because N could be large. Eshenauer and Gligor proposed in [18] a random key pre-distribution scheme: random sets of keys are distributed to each sensor and after deployment, any pair of nodes has at least one shared key to use as their secret pairwise key. Chan et al. in [19] proposed a q-composite random key pre-distribution scheme that improves the resilience of the network compared to the Eschenauer-Gligor scheme. The difference is that q common keys – instead of just one – are needed to establish secure communications between a pair of nodes. The secret shared key is the hash of the q common keys.

Liu et al. proposed in [20] a key pre-distribution scheme that relies on location deployment knowledge in order to improve the probability of key sharing. The keys are assigned according to the geographical position of sensor nodes. A similar approach is also developed in [21]. Such solutions are impractical in sensor networks with randomly deployed topology. A polynomial based key pre-distribution scheme is proposed in [22]: a polynomial share is distributed to each node and using it, any two nodes are able to establish a pairwise key.

Perrig et al. proposed in [23] SPINS, a key management protocol that relies on a trusted base station to distribute keys. Two sensor nodes use the base station as a trusted third party to set up their pairwise secret key. SPINS includes two parts: SNEP (Secure Network Encryption Protocol) that secures communications between a node and the base station or between two nodes, and μTESLA (μTime Efficient Streaming Loss-tolerant Authentication) that authenticates packets coming from the base station.

Nevertheless, symmetric key based schemes are only applicable to legacy sensor nodes, which are seen as belonging to sensor networks, themselves connected to the internet via dedicated gateways. These schemes are based on pre-shared keys between different nodes within the same sensor network. In view of the IoT scenarios, however, a sensor node is considered as a part of the Internet able to establish end-to-end communications with external entities without requiring any initial knowledge of these external entities or any prior authentication context or pre-shared keys. In terms of security, these schemes rely on link-layer security and are especially vulnerable to node compromise. Besides, symmetric algorithms offer poor authenticity and data integrity services since Message Authentication Codes (MACs) are not publicly verifiable. Scalability and complex key management remain also significant issues considering a large-scale sensor network which needs to generate a huge number of shared keys and then install them in sensors before their deployment.

In order to eliminate the complexity of key management and increase the security level within the sensor network, many researchers investigated the application of asymmetric cryptography to sensor networks in order to provide the best trade-off between security services, computation overhead, and memory requirements. The security services (e.g. non-repudiation) and protection level (e.g. resilience to node compromise) it offers are more evolved than those offered by symmetric cryptography. Lopez in [24] highlights the limits of using symmetric cryptography in sensor networks and promotes

solutions based on public key cryptography to enhance the security of the entire system, while warning against complex computations. The author emphasizes the important role that Elliptic Curve Cryptography (ECC) can play to overcome the computational complexity of public key algorithms. ECC was the top choice among different public cryptography algorithms due to its lower energy consumption, fast processing time, compact signatures, and small key size. For example, a 160-bit ECC key size guarantees a level of protection equivalent to a 1024-bit RSA key, with an energy consumption reduced by half [25]. The authors of [26] present lightweight implementation of public key cryptography algorithms relying on elliptic curves and claim that using ECC-based key establishment solution is the best trade-off between energy consumption and security level.

Alternatively [27], [28] focus on making the well-known RSA public key cryptosystem [29] more adapted to resource-restrained devices using a small RSA public exponent (e) and a short key size. For example, Watro et al. in [28] develop the TinyPK system that allows implementation of PKI in sensor networks. The concept requires the use of smaller RSA parameters (key size, exponent) and the use of public key operations only at the sensor device. This comes, however, at the price of a lower security level [32]. Huang et al. [30] and Kotzanikolaou et al. [31] propose hybrid protocols that combine standard Elliptic Curve Diffie Hellman (ECDH) key agreement and implicit certificates with symmetric techniques in an effort to reduce the expensive elliptic curve random point scalar multiplications at the sensor side. The cost per node for key establishment is effective due to the combination of symmetric encryption in the randomisation process and the use of Schnorr signatures. This approach reduces the high cost of public key operations by replacing asymmetric-key operations with symmetric-key based ones and thus joins the advantages of both approaches. However, communications with an external party become less feasible, since both peers have to share a symmetric key.

To make public key cryptography practical in WSN, [32] [33] [34] have proposed hardware solutions that extend computational capabilities of a standard node through low power hardware modules. Results show that these additional hardware implementations help to provide the security services with less energy consumption and at a low cost; however, it could be a hard task taking into account the cheap and small design of sensor devices

With the wide deployment of WSN applications, a different model of sensor networks, named Heterogeneous WSN (HSN), has emerged over the last several years. Contrary to the homogeneous sensor network, in heterogeneous networks different sensors with different capabilities, sensing for different applications coexist in the same monitored environment.

Accordingly, Mache et al. developed in [35] a hybrid key establishment framework for resource-restrained sensor networks that exploits heterogeneity of sensor node deployment, basing on a combination of symmetric and asymmetric operations. The idea is to use less expensive symmetric cryptography on the first part of the path from sensor to sink until a resource-rich gateway is reached, and then to use more expensive public-key cryptography on the second part of the path.

Riaz et al. proposed in [36] three key establishment schemes: SACK based on symmetric key cryptography, SACK-P based on asymmetric key cryptography and SACK-H which relies on a hybrid cryptography approach using asymmetric cryptography for cluster-wide communication and symmetric cryptography for network-wide communication. The authors then draw a comparison between the three proposed schemes and show that SACK is light on resource consumption but provides a low security level since one node compromise makes the whole network vulnerable. In contrast, SACK-P is heavy on resource consumption but provides the highest security level with a maximum resilience to node compromise. The hybrid scheme SACK-H falls between the two others and presents medium resource consumption with a medium security level.

However, security is provided in these schemes on a hop-by-hop basis. Confidentiality and availability are thus compromised since the intermediary translating entity at border between “symmetric” and “asymmetric” domains, introduces potentially both a security flaw and a single point of failure.

1.4.2.2. Towards secure integration of IP enabled WSNs with the Internet:

The solutions reviewed above for key establishment in traditional WSNs are not targeting a secure end-to-end communication between the sensor node and remote hosts. Instead, they discuss security of

communications within the sensor network. Recently, with the advent of WSNs integration to the Internet, the need for an end-to-end security protocol between sensor nodes and the legacy Internet has been recognized. In order to enable functional implementations of TLS and IPsec in a constrained environment, lightweight key establishment schemes have been proposed. They base mainly on the use of modified implementations of the corresponding keying protocols: TLS handshake, IKE and HIP BEX.

1.4.2.2.1. Lightweight TLS handshake proposals

i. Basic TLS handshake

When a TLS connection is needed between a client and a server, an initial phase called TLS Handshake [9] is needed to negotiate security algorithms, to authenticate at least one peer to the other and to establish a shared secret between both peers. The TLS Handshake protocol supports two different key exchange methods: a key transport method based on RSA asymmetric cryptography and a Diffie-Hellman key agreement method. The entire exchange is illustrated in the figure 4.

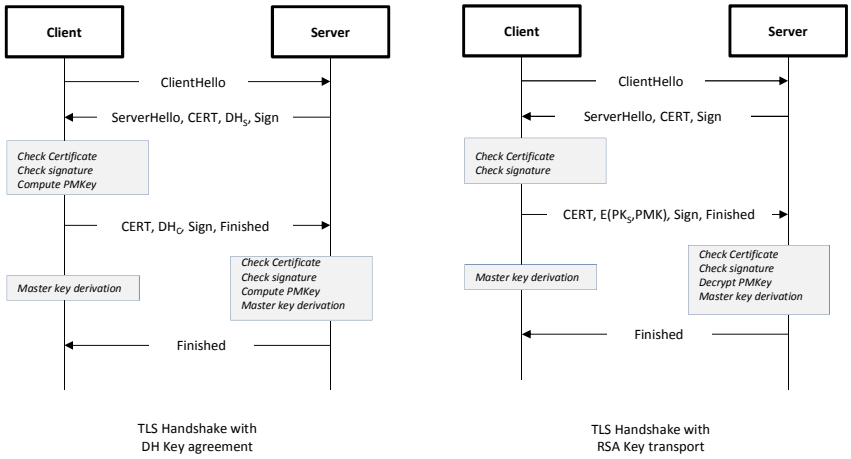


Fig. 4. Basic TLS handshake with two supported key delivery modes.

First the client and the server exchange *Hello* messages. These messages contain nonces and negotiate the set of cryptographic algorithms that will be applied to the session. The server Hello also contains the server's Diffie–Hellman public value if a DH key agreement is performed, along with the server certificate and a signature for authentication.

Next, the client sends a message containing either its Diffie–Hellman public value in case of a DH key agreement or a generated secret – called pre-master key (PMK) – if a key transport method is performed. In this latter case, TLS handshake performs the one-pass key transport so that the pre-master key is pushed from the client to the server. Indeed, the assumption that the server's certificate can be validated by the client sounds more realistic than the opposite; this assumption actually laid the bases for today's secured HTTPS transactions. The PMK is thus encrypted using the server's RSA public key, which is retrieved from the server's certificate. This message also includes a signature on the hash value of the PMK, combined with all past messages exchanged during the current session. The client authentication is optionally performed too during TLS handshake: if requested by the server, the client provides it with a certificate and a signed message.

The client and the server can then retrieve the shared pre-master key using the selected key exchange method. That is, each can compute it as the Diffie–Hellman shared secret derived from the two exchanged DH public values, or the server can decrypt the encrypted secret pushed by the client using its RSA private key. In order to reduce the PMK storage requirement at the communicating parties and to ensure the key freshness, a master secret is derived from PMK using a hash function applied to the concatenation of the PMK and the two nonces exchanged in *Hello* messages.

The Finished message ends the handshake exchange. It includes a hash computed over the master key and all the past messages. The receiving entity is able to compute the corresponding hash value from its own records in order to check if the result matches the received value.

ii. Lighter TLS handshake Declinations

It is worth noting first that using pre-shared keys for key exchange as in TLS-PSK [37] cannot be practical between IoT nodes as explained before, due to the absence of initial authenticating context between them.

As explained above, the use of ECC was generally considered to be the most suitable choice among other public key cryptosystems in legacy WSNs. Accordingly, we have identified two different lightweight implementations of TLS on constrained devices that base on ECC during the key exchange while maintaining the same message exchanges. Sizzle [38] was the first security protocol that proposed the use of TLS in the WSN in order to implement an HTTPS stack. Sizzle relies on translating gateways that map the sensor nodes local (non-IP) addresses to internet hosts IP addresses, allowing them to exchange data directly with remote IP peers. During the TLS handshake, the Elliptic Curve Diffie-Hellman (ECDH) key agreement [39] and the Elliptic Curve Digital Signature Algorithm (ECDSA) [40] respectively replace the Diffie-Hellman key agreement and DSA algorithms. Using these ECC-based protocols, performance evaluations showed that implementing HTTPS web servers on sensor nodes may be supportable for infrequent connections. In 2009, SSNAIL [41] has been developed as a second lightweight TLS implementation for IP-based WSNs relying on the same cryptographic primitives as Sizzle for the key exchange while eliminating the use of the gateway. Authors measure that implementing an ECC-based full handshake takes around 1 second while it takes 8.5 seconds for an RSA-based one.

1.4.2.2.2. Lightweight IKE proposals

i. Basic Internet Key Exchange

The objective of IKE [12] is to establish a secure channel between two parties and enable them to mutually authenticate each other. IKE provides a protocol to establish security associations (SAs) that are needed to secure IP datagrams using IPsec:

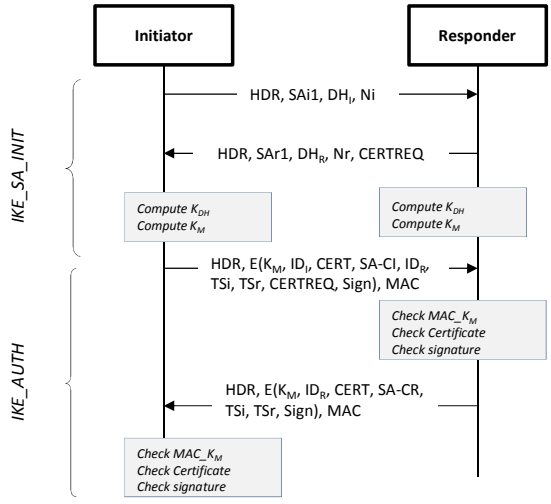


Fig. 5. Basic Internet Key Exchange (Establishment of a simple SA).

All IKE communications are in the form of request-response pairs. An IKE transaction consists of two required request/response exchanges, as depicted in figure 5. The first request/response exchange (IKE_SA_INIT) negotiates cryptographic algorithms (SAi1, SAr1), exchanges nonces (Ni, Nr) and performs the Diffie-Hellman exchange to establish a shared key. The messages in this exchange are not authenticated; the following exchanges authenticate these messages by including their content while calculating the authentication values. At this stage, both sides have enough information to set up a master key K_M, using both Diffie-Hellman public values and the nonces. All shared keys for the IKE SA are then derived from this master key.

The second request/response exchange (IKE_AUTH) authenticates the previous messages. The identities of both sides are authenticated, and a simple IPsec SA, called a child SA, is established. Security association descriptions (SA-CI, SA-CR) indicating the supported cryptographic algorithms

and the traffic selectors (TS_i, TS_r) are exchanged. Parts of these messages are encrypted and integrity protected (with a MAC) using the master key established in the IKE_SA_INIT exchange.

At this stage, the IKE transaction has been authenticated and a single child SA has been established. If no other child SAs are required, the IKE transaction terminates here. If, however, additional child SAs are required, the transaction moves to create another child SA.

ii. Lighter IKE Declinations

In 2011, a first compressed IPsec implementation for 6LoWPAN networks has been proposed [42] , basing on pre-shared keys for key exchange. Authors recognize that using pre-shared keys is not a feasible solution since sensor nodes should be able to communicate with external hosts without the need for prior authentication contexts. They are currently investigating the feasibility of Internet Key Exchange of IPsec for 6LoWPAN.

Independently from the integration of WSN with the Internet, two recent variants of IKE have been proposed for energy efficiency purposes. V. Nagalakshmi in [43] modifies the IKE protocol by eliminating pseudo random generation functions, thus eliminating its repetitive usage during the key exchange. The sender transmits a hash of its private key and its Diffie-Hellman private value instead of sending nonces. The proposed work leads to cost effectiveness, however, the energy cost of a pseudo random function generation (amounting to a symmetric encryption) can be neglected compared with the heavy cost of asymmetric cryptographic operations that are required further in the protocol exchange. In 2012, an ECC-based IKE protocol [44] has been designed for Internet applications. It aims to reduce the heavy burden of the base exchange of the protocol IKE by using ECDH key exchange to set up the shared key and using ECC-based public key certificate for the authentication of the communicating entities.

1.4.2.2.3. Lightweight HIP BEX proposals

Like IKE, HIP BEX aims at generating key material for a subsequent use by IPsec in order to establish a secure end-to-end communication between two entities. However, contrary to IKE, no certificates are required in HIP BEX for the authentication, because self-certifying identifiers are used. The concept of a "self-certifying identifier" can be explained as follows: it is an identifier that only the legitimate owner can use, without needing an external proof coming from a trusted third party (certificate) to claim its ownership. In order to achieve this functionality, the self-certifying identifier is generally built in the form of a "cryptographically generated identifier" (CGA) [45]. This latter must be univocally bound to a public key, whose private counterpart is only known to the legitimate owner, hence its denomination. Thus, proving the ownership of a certain CGA amounts to proving the ownership of the related public/private key pair, hence the ability to use the corresponding private key. In HIP, the CGA used to identify a node is the Host Identity Tag (HIT), which is a 128-bit hash of its public key.

i. HIP Base Exchange (BEX)

The objective of the HIP Base Exchange (BEX) [13] is to perform authenticated key agreement between two HIP peers. The entire exchange is depicted in figure 6.

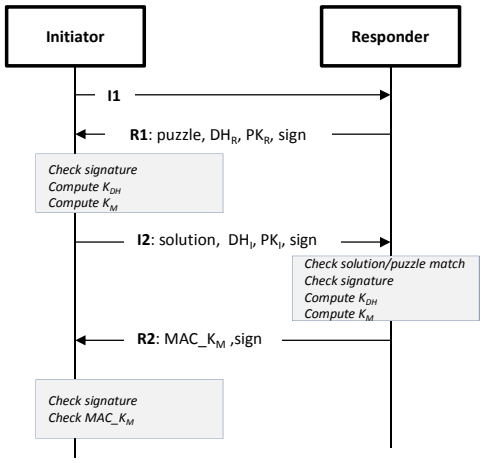


Fig. 6. HIP Base Exchange (BEX).

The message I1 initiates the exchange. This message only includes the initiator and responder identities (HIT_I, HIT_R). Upon reception of I1, the receiver sends a (possibly pre-computed) message R1 composed of a puzzle, its Diffie-Hellman public value, its public key (or Host Identifier) and a signature. The initiator has to answer this message with an I2 message, composed of the puzzle solution (so as to prevent DoS attacks), its own Diffie-Hellman public value, its own (possibly encrypted) public key and a signature.

At this stage, the initiator and the responder are able to compute the Diffie-Hellman shared key and derive the master key as the hash value of this pre-master key concatenated with the two peers' identifiers and a nonce.

Finally, with the last message R2, the responder finalizes the exchange. This message includes a HMAC computed using the DH shared key, and a signature.

ii. Lighter HIP Declinations

Stemming from the observation of the heavy computational cost of HIP Base Exchange, two modifications of HIP have been proposed in order to make the protocol usable by constrained nodes.

HIP Diet Exchange (DEX) [46] proposes that a node use a long-term Elliptic Curve Diffie Hellman (ECDH) public value as its Host Identifier. DEX then adapts the key exchange, as depicted in figure 7.

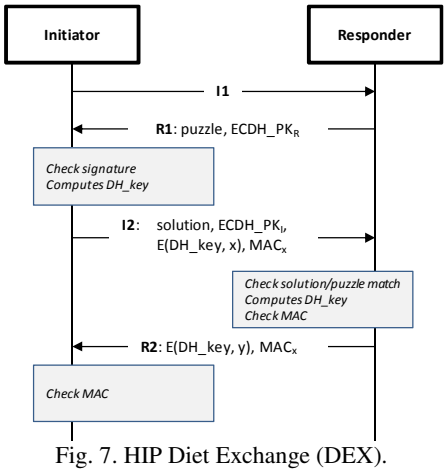


Fig. 7. HIP Diet Exchange (DEX).

The Host Identifier being itself the Diffie-Hellman public value, there is no need to authenticate it through asymmetric cryptography. The knowledge of the DH key is enough to prove that a node is a legitimate peer in the exchange. Accordingly, this DH key is used to transport two random seeds *x* and *y* that are eventually used to derive the final secret.

As compared with HIP BEX, the single computation of the long-term Diffie-Hellman public values eliminates the DH key generation cost. Likewise, the use of Elliptic Curve Diffie-Hellman and the fact that no other asymmetric cryptography operation is required make the key exchange lighter.

Lightweight HIP (LHIP) [47] is a much more radical approach, which keeps the same message syntax as in HIP BEX for compatibility reasons but does not use any of the HIP BEX security mechanisms. No Diffie-Hellman key is computed, no RSA operation is performed and no secure IPsec tunnel is set up after the exchange. Instead, hash chains are used to cryptographically bind successive messages with each other, which represents a minimal degree of security. LHIP procedure is depicted in figure 8. Note that the DH_R, DH_I, PK_R and PK_I message fields are present in the exchange but are unused in standard LHIP exchange except when upgrading to standard HIP BEX, which LHIP allows.

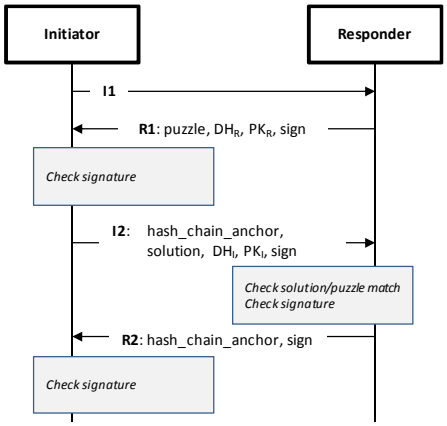


Fig. 8. Lightweight HIP (LHIP).

LHIP trades security for energy efficiency in a drastic manner. Its security level is therefore very low: only HIP control messages (e.g. supporting node mobility) are integrity-protected through hash chains. This weak security property only guarantees that an ongoing session has not been hijacked (temporal separation property) but does not provide strong node authentication. Besides, HIP data messages are not protected since no key exchange mechanism is provided.

1.4.2.3. Discussion

As presented above, most of modified variants of TLS handshake, IKE and HIP BEX rely on the use of ECC algorithms. In [48], a comparative performance analysis has been conducted between RSA-based and ECC-based TLS handshakes on a standard PC node. Results have shown that using ECC reduces by 37 percent the energy consumption of the TLS key establishment process compared to RSA. Liu et al. in [49] implemented ECC in TinyOS for many platforms including MICAz and TelosB. They assessed the ECC (160-bit keys) point multiplications cost needed to perform the ECDH exchange and ECDSA signatures. Results have shown that the energy cost of ECDH-ECDSA key agreement protocol is around 236 mJ for MICAz and 72 mJ for TelosB. Based on energy costs of Table 4, a DH-RSA key agreement protocol consumes around 190 mJ on a TelosB. Hence, the energy consumed with the use of ECC is reduced by 62 percent.

Nevertheless, these measured energy costs of ECC are still non negligible, being in the order of magnitude of millijoules. In practice, these energy costs would be hindering for highly resource-constrained nodes in the IoT. Authors in [63] investigate the practical use of ECC on constrained devices in WSNs and conduct a cost comparison between two key establishment schemes ECDH-ECDSA and Kerberos (a server-assisted key distribution protocol based on symmetric cryptography). They conclude that Kerberos is 95 times less costly than ECDH-ECDSA on a MICAz sensing platform.

This unsuitability of prior key establishment proposals for constrained devices accentuates the need for novel IoT-specific solutions. This need was recognized in [51] and left open. According to the literature, the design of an efficient key establishment approach for existing security standards that clearly addresses the heterogeneous IoT communications has not been undertaken yet [51]. Further careful design is required to reduce the energy cost of key establishment schemes while taking into account the heterogeneous nature and the end-to-end security requirement of the IoT.

1.5. CONCLUSION

In this chapter, we addressed the new security requirements of the Internet of Things. This promising paradigm aims to the integration of several architectures and the support of new communications between heterogeneous nodes, commonly reachable over a global IP-based infrastructure, in spite of having highly distinct characteristics. In order to securely accomplish this integration, end-to-end communications have to be established. IoT nodes require therefore the ability to set up a shared secret between one another, in order to bootstrap secure communications. Adoptability of existing security protocols is an important requirement for an IoT key establishment protocol, since the IoT will encompass today's Internet and may not be based on clean slate approaches. However, straightforwardly reusing existing schemes cannot be feasible because of the efficiency requirements of the IoT. Existing key establishment protocols involve heavy cryptographic operations that resource-constrained IoT components cannot support. In the literature, the design of efficient key establishment protocols that clearly address heterogeneous IoT communications is not undertaken yet.

A first section reviewing existing key establishment schemes was essential in this chapter in order to reason on how to efficiently adapt them to the IoT scenarios. We provided a classification of key establishment protocols according to three criteria: the key delivery scheme (key transport or key agreement), the underlying cryptographic primitive family (symmetric or asymmetric) and the authentication method. Considering the initial requirements of the IoT, we have retained TLS handshake, Internet key Exchange and HIP BEX protocols as the best candidates. However when assessing them in terms of energy efficiency, we have illustrated the heavy computational cost they require to run on constrained devices. In the literature, energy efficiency was an important concern in WSNs because of the low capabilities of sensor nodes. Unfortunately IoT requirements go far beyond those of WSNs, since it is assumed in these latter that the sensor nodes are isolated from the internet and connected to external hosts via dedicated gateways. The few works focusing on making lighter the retained key establishment schemes proposed to replace the heavy cryptographic operations of RSA and Diffie-Hellman algorithms with the use of Elliptic Curve Cryptography. However, recent studies have proved that the energy costs of ECC are still non-negligible when implemented on highly-constrained devices. In the second chapter, we take into account the inadequacies of these proposals as well as the identified requirements for suitable IoT key establishment schemes for designing new keying solutions able to enable end-to-end secure communications between nodes with different resource capabilities, in the context of IoT.

Chapter 2: COLLABORATIVE KEY ESTABLISHMENT

2.1. INTRODUCTION

In the previous chapter, we highlighted how heterogeneity of nodes and communications in the Internet of Things brings new security challenges that have to be considered for the design of further security solutions. In this chapter, we tackle heterogeneity from a different axis, trying to take advantage of it to design our solution for IoT key establishment.

Spatial heterogeneity is frequent in the IoT as long as different nodes with different resource capabilities acting for different services coexist within a global unified architecture. Heterogeneity can also evolve over the time when considering other factors such as the mobility of nodes or the dynamic changes in the amount of available resources (resource exhaustion, resource harvesting). Bearing in mind this heterogeneity aspect, the main rationale of our solution is to make a highly resource-constrained node able to establish secure contexts with other unconstrained nodes within a heterogeneous IoT architecture. We explored the possibility of reducing the computational load to be performed on constrained devices instead of only thinking on reducing the cost of cryptographic primitives, as proposed before. Eventually, we proved that we can exploit heterogeneity of nodes in order to offload heavy computational operations required at the constrained device to more powerful nodes in the surroundings.

Accordingly, we proposed to redesign existing key establishment schemes so that the constrained peer may delegate its heavy cryptographic load to less constrained nodes in neighbourhood. During the key exchange, these assisting nodes, or "proxies", take charge of the session key derivation, in a collaborative and distributed manner. However, the session key is known only by the two endpoints of the communication, in order to guarantee its secrecy. Several constraints have been considered in the design of our approach: (i) the collaborative scheme must not come at the expense of a key disclosure risk or a collusion attack (ii) in case of a proxy unavailability or a greedy behaviour, the system should continue to run properly (iii) each proxy is required to prove its legitimacy by proving that it is authorized by the constrained node to act on its behalf.

We start this chapter in section 2.1 with a description of the prerequisites for our collaborative keying solution. Network model, assumptions and initial operations for bootstrapping assisting entities are presented in that introductory section. Section 2.2 then details the proposed approaches: two novel collaborative algorithmic key establishment protocols are introduced, respectively for key transport and key agreement. These approaches are mapped in section 2.3 with the key establishment communication protocols identified in previous chapter as relevant for the IoT. Instantiations of the collaborative approach are therefore proposed as updated versions of the IKE, TLS and HIP protocols. Assessments of these updated protocols, respectively in terms of efficiency and security are proposed in next sections 2.4 and 2.5, which respectively address performance evaluation and formal security analysis. Finally, section 2.6 concludes this chapter.

2.2. REQUIREMENTS AND BOOTSTRAPPING

2.2.1. Considered network model

Our network model is deduced from the paradigm we envision: we consider a global IoT infrastructure that interconnects heterogeneous nodes with different capabilities in terms of computing power and energy resources. Among these heterogeneous nodes, we especially consider three different categories:

- Highly resource-constrained nodes, unable to support the computational cost of asymmetric cryptographic operations required by the key exchange phase, while nevertheless requiring end-to-end security (e.g. sensor nodes).
- Proxies at neighbourhood, less constrained and therefore able to perform cryptographic operations. These nodes may either be dedicated assisting servers or nodes belonging to the same local infrastructure, though being less impacted by energy constraints (e.g. having energy harvesting capability).
- Unconstrained nodes, not belonging to the same local infrastructure, with high energy, computing power and storage capabilities (e.g. line-powered remote servers).

The considered scenario in this thesis can be summarized as follows: a highly resource-constrained sensor node (the source node A) needs to exchange sensitive data with an external server (the destination node B) on an end-to-end basis. These two entities are supposed to have no prior knowledge of each other and no prior shared key. Initially, their objective is therefore to setup a session key with each other. This scenario is likely to occur if one considers an IP sensor node (e.g. 6LoWPAN sensor) that has to deliver sensitive sensed data to remote peers with which it has not yet established shared secrets. This delivery may either happen through a *pull* model, wherein the sensor (IoT resource) is explicitly requested to provide data by a remote IoT requester, or through a *push* model, wherein the sensor is intermittently sleeping and regularly wakes up in order to push sensed data towards a (configurable) set of peers.

2.2.2. Assumptions

1. After the initialisation phase, every sensor node shares pairwise keys with a subset of its one-hop neighbours. These keys may have been generated during a specific bootstrapping phase using a trusted key management server or through more subtle mechanisms such as transitive imprinting[8] [52].
2. The highly resource-constrained node is able to identify a set of less resource-constrained nodes that are available for supporting heavy cryptographic operations on its behalf. The identification process is detailed in the fourth chapter of this thesis.
3. There exists a local trusted entity within the sensor network that owns a shared secret with all nodes in the sensor network and a public/private key pair.
4. The external server does not communicate with the sensor network trusted entity but is statically configured with or able to validate its public key.

The considered network model and assumptions are represented on figure 9.

[8] With bilateral imprinting, physical devices establish shared secrets with one another through the use of a dedicated short-range wireless transmission such as NFC. In order to resolve the problem of user-interaction scalability, transitive imprinting is introduced to allow two devices to establish a secret key based on an intermediate device with which both have already secure associations.

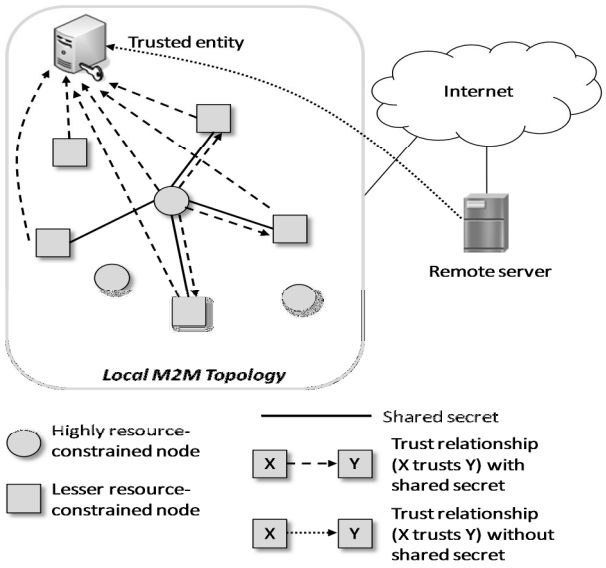


Fig. 9. Network model and assumptions.

2.2.3. Preparation of the involved entities

As an initial phase, the resource-constrained sensor node A carefully selects the $P_1 \ldots P_n$ proxies that will assist its key exchange. This operation is based on the trust management system that is presented in the fourth chapter of this thesis.

Our approach requires that the $P_1 \ldots P_n$ nodes process messages on behalf of the resource-constrained node during the key exchange. Hence authorisation and authentication questions arise at the proxy sides, since these nodes should be provided with a representativeness proof. This proof could be a certificate including the proxy's public key associated with the right "authority to sign on behalf of A", all of which signed with the source's private key and delivered 'offline' to the proxy, regardless the current exchange. However the use of long-time authorisation certificates could be diverted for malicious exploits.

Hence, the certificate should include other dynamic parameters added by the source node in order to restrict the ability of proxies to act on its behalf, such as the identity of the destination node, a session nonce, or an expiration date. In this case, the authorisation proof should be delivered 'online' to the proxy during the protocol exchange. Nevertheless, managing dynamic certificates would be hindering for the constrained sensor node.

For this reason, we propose to move the computational load required to dynamically manage authorisation proofs from the sensor node to a local, unconstrained, trusted entity T (in a sensor network T can be the base station), which will be the only entity able to assert that a proxy node is authorized to sign on behalf of A. On the other hand, the verification of each proxy's certificate would be also heavy for the destination node. We propose therefore to rely on the technique of authenticated dictionaries such as Merkle tree [53] or one-way accumulators [54] in order to efficiently authenticate participants and validate their membership to the group of selected proxies at the server side.

A **Merkle tree** structure provides a means to authenticate a high number of items without individually signing each of them, but rather authenticating them as a whole. In a nutshell, the items to authenticate are placed in the leaves of a binary tree. The item corresponding to a parent node is computed from the items of its two children, e.g. through a one-way hash function. Eventually, all leaf items are involved in the computation of the root node value. Thus, only this value has to be authenticated in order to authenticate all items of leaves. The membership of a leaf in the group can then be verified with respect to a publicly known root value and its authentication path, this latter being defined as the successive items required to compute the root value from the considered leaf.

Using this technique, the destination node has thus only to verify once the signature of the root value to authenticate all proxies public keys. The process is bootstrapped as follows. From the public

keys of the selected nodes $P_1 \ldots P_n$, T can securely provide each proxy P_i with its authentication path MT_{Path_i} in the Merkle tree of all *n* public keys, along with a T-signed message consisting of:

- MT_{Root} , the root of the Merkle tree of all *n* public keys;
- An anti-replay nonce;
- R_P (Reconstitution Parameters), the number of proxies sought to participate in the key establishment process along with the minimum number of cooperative proxies required to recover the original message;
- A's identity, which will make B aware of the node obtaining assistance from the proxy P_i.

One-way accumulators are another technique of authenticated dictionaries. One-way accumulators are based on one-way hash functions which satisfy a quasi-commutative property. Thanks to this property, items of a group (x_1, …, x_n) agree on accumulated hash of their values $y = H(x_1, \ldots, x_n)$ and each item keeps this hash function H, its own value x_j and an accumulated hash y_j for all other items of the group $x_{i \neq j}$. To prove its membership, it needs to present the pair (x_j, y_j) in order for the recipient to verify that $H(x_j, y_j) = y$.

Here again, using this technique, the destination node has only to verify once the signature of the accumulated hash for all proxies of the group instead of validating the signature of each proxy's certificate apart. The corresponding process in bootstrapped as follows. From the public keys of the selected nodes, T can securely provide each proxy with an accumulated hash of all other participants public keys. A proxy P_i will thus be provided with $H(K_{P1},\ldots,K_{Pi-1},K_{Pi+1},\ldots,K_{Pn})$ with H() being a commutative one-way hash function, along with a T-signed certificate consisting of:

- $H(K_{P1} \ldots K_{Pn})$, an accumulated hash of all *n* public keys;
- An anti-replay nonce;
- R_P (Reconstitution Parameters), the number of proxies sought to participate in the key establishment process along with the minimum number of cooperative proxies required to recover the original message;
- A's identity, which will make B aware of the node obtaining assistance from the proxy P_i.

Upon receiving their proof material, proxies are prepared to participate to the collaborative process.

2.3. KEY EXCHANGE DESCRIPTION

In order to first give a clear description of the proposed collaborative process, we deal independently in this section with each of the two *key exchange algorithmic protocols* that were identified as highly relevant in the first chapter, namely key transport and key agreement. Then, in the next section, we modify the retained *key exchange communication protocols*, namely TLS Handshake, IKE and HIP BEX by applying these collaborative key exchange schemes.

2.3.1. Collaborative key transport

In this subsection, we describe how we offload the key transport computational load from a highly constrained node to a set of proxies. We consider first the one-pass key transport mode and then adapt the proposed solution to the two-pass key transport mode.

2.3.1.1. Collaborative one-pass key transport

In a standard one-pass key transport mode, a random secret key x is generated by the source A and securely delivered to the node B.

The objective for the highly resource-constrained node A in the collaborative one-pass key transport mode we propose is to generate a random secret key x and then to rely on a set of proxies to deliver it to the server B, using asymmetric cryptography. We propose two techniques to distribute the

computations required for the secret key delivery. The successive phases that make up our proposal are illustrated in figure 10 below, and explained later in the following subsection.

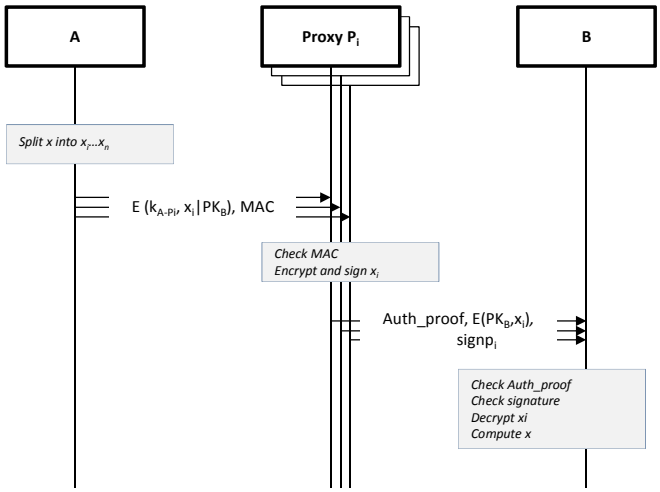


Fig. 10. Collaborative one-pass key transport.

2.3.1.1.1. Simple secret partition

A starts by splitting the secret x into *n* parts $x_1,\ldots,x_n$ with $x=x_1|x_2|\ldots|x_n$ and then securely sends each part x_i to the corresponding proxy P_i. The part of the secret x_i is transmitted encrypted with the shared key between A and the proxy P_i (see assumption 1).

Upon reception of the x_i secret key part, the proxy P_i encrypts it using the server's public key and signs the result using its private key.

We propose to use the lightweight one-time signature scheme of Lamport [55] in order for the proxy to sign messages on behalf of the constrained node. This signature scheme is especially lightweight and computationally efficient compared to other signature schemes [56]. Two drawbacks could possibly mitigate its practical applicability: on one hand, a public/private key pair should be used only once since information about the private key is divulged along with the signature itself. On the other hand, a long key will be needed to sign a long message, since the private (resp. public) key is the concatenation of all private (resp. public) values, as numerous as the message blocks and being each as long as the associated hash function output. Nevertheless, neither of these shortcomings affects our approach, which addresses one-time exchanges of short messages. In this case, we propose that T generates the Lamport private/public keys for each proxy P_i and securely provides it with this key material along with the authorisation proof of subsection 2.2.3, in the same message.

After receiving the required key material, the proxy signs the encrypted secret x_i and then sends the result to the server B. In turn, B verifies the integrity of the received message using P_i's public key and eventually decrypts x_i.

We assume that each proxy P_i has initially contacted B in order to request its certificate and to provide it with its own proof material. In response, after verifying the signature of T (see assumption 4), B verifies that the proxy has supplied a valid public key and that it is a valid proxy assisting A in its key establishment process. Having received all x_i fragments, B becomes able to recover the original secret key x.

2.3.1.1.2. Threshold secret distribution

At this stage, it is worth noting that the solution proposed above is based on the reliable deliveries of all secret fragments x_i in order to be able to reconstitute the source secret key at the destination node. A single missing message from a proxy makes the information incomplete for the server and may[9] fail the protocol exchange.

[9] The protocol might be resumed, if one assumes that it implements an acknowledgement/retransmission mechanism for fragments delivery. In terms of state machine complexity and bandwidth inefficiency, this may not fit however to the envisioned highly-constrained client nodes.

Yet, assuming that proxies behave as honest and reliable participants could be difficult in practice: even in scenarios where dedicated trustworthy proxies are made available to resource-constrained nodes, reliability of those proxies is not guaranteed. In order to reinforce the reliability of the proposed distributed scheme for one-pass key transport mode, we rely on a threshold secret distribution wherein a forward error correction scheme [57] is applied by the source A to the secret x, in order to handle losses and missing secret parts from assisting nodes.

The principle of forward error correction scheme is to add redundant parity packets to the original message, divided into multiple packets, in order for it to be recovered by the receiver even if some packets were altered or lost during the process of transmission. Let n be the total number of sent blocks, k ($k<n$) is the minimum number of blocks required to reconstruct the original message.

First, the source node performs the split process of the secret key. Then it applies the error redundancy scheme to the fragments of the secret key as depicted in figure 11 below.

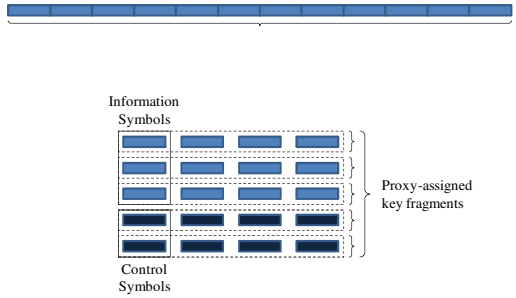


Fig. 11. Adding redundancy for reliable one-pass key transport.

Hence, the server B becomes able to reconstruct the session key provided that a sufficient number of packets from assisting nodes are received, without requiring the reception of all of them. This technique protects our solution from unreliable delivery in proxy→server connection, though the source node should perform more computational operations in the initial phase, to compute the redundant packets.

In addition to the protection against packet loss, the threshold approach can protect our solution against malicious proxies. A node incorrectly processing a conveyed fragment of the secret key (e.g., replacing the received fragment with a forged one before delivering it to B) can be identified at the server side. Indeed, this latter can compute different combinations of k messages from the pool of n messages and detect the node providing wrong information. We give the example below to explain how the cheater detection process can take place within the threshold approach.

Let 5 be the total number of proxies and 3 the minimum number of packets required to reconstruct the original secret at the server side. We consider that proxy 2 is a malicious node transmitting bogus data instead of correctly encrypting its corresponding fragment of the secret key. The server decrypts the received messages from proxies and combines the resulting key for each l-uplet ($k \leq l \leq n$) of received messages as follows:

Table 5: Malicious proxy identification and key retrieval through multiple l-uplet processing.

l-uplet	obtained key	l-uplet	obtained key
{1, 2, 3}	α	{2, 4, 5}	η
{1, 2, 4}	β	{3, 4, 5}	**δ**
{1, 2, 5}	γ	{1, 2, 3, 4}	θ
{1, 3, 4}	**δ**	{1, 2, 3, 5}	ι
{1, 3, 5}	**δ**	{1, 2, 4, 5}	κ
{1, 4, 5}	**δ**	{1, 3, 4, 5}	**δ**
{2, 3, 4}	ε	{2, 3, 4, 5}	λ
{2, 3, 5}	ζ	{1, 2, 3, 4, 5}	μ

With these intermediary results it would be possible for the server to learn that the proxy 2 is the cheater element of the group, since the same (correct) value δ is obtained for the key whenever the fragment retrieved from the proxy 2 is not used during the session key computation.

2.3.1.2. *Collaborative two-pass key transport*

In a two-pass key transport mode, a random secret key x generated by the source A and a second random secret value y generated by the server B are securely exchanged between A and B and used to compute the session key. As explained above, it is safer to involve both parties in the session key derivation compared with what happens in the one-pass key transport mode where the secret key is entirely controlled by only one partner. The phases of the proposed solution are depicted in figure 12 below.

We propose to apply the same collaborative approach as described in the one-pass key transport scheme to deliver the secret x from the source to the server. After having received a sufficient number m ($m > k$) of x_i fragments, the server obtains the secret value x. At this stage, it generates in turn a secret key y to be provided to the resource-constrained client. However, this latter cannot decrypt and verify the integrity of the received value because of its resource constraints. For this reason, we propose that the proxies support also the reception of the secret key y on behalf of A in a cooperative manner. That is, these nodes take charge of the computational load required to decrypt and verify the received message from the server and then transmit it securely to the source. Yet, the divulgation of the secret key y to the proxies would affect the security of our system. In order to preserve the secrecy of y, we propose to have it encrypted with the secret key x reassembled by the server in the previous step. The x-encrypted secret key y is MACed with the secret x and then signed with the server's private key. It is finally sent to each proxy P_i, which has to verify the integrity of the received packet from the server before decrypting it. Then the packet content (that is, y encrypted and MACed with x) is securely transmitted to the client. As long as an appropriate number of the same packet is received from different proxies, the client ensures the validity of the transmitted message from the server. Consecutively, it checks the MAC in order to ensure that the server has obtained the same secret x and verify the message integrity. Once the client A receives a valid message, it can obtain the transmitted secret value y in order to complete the set-up of the session key.

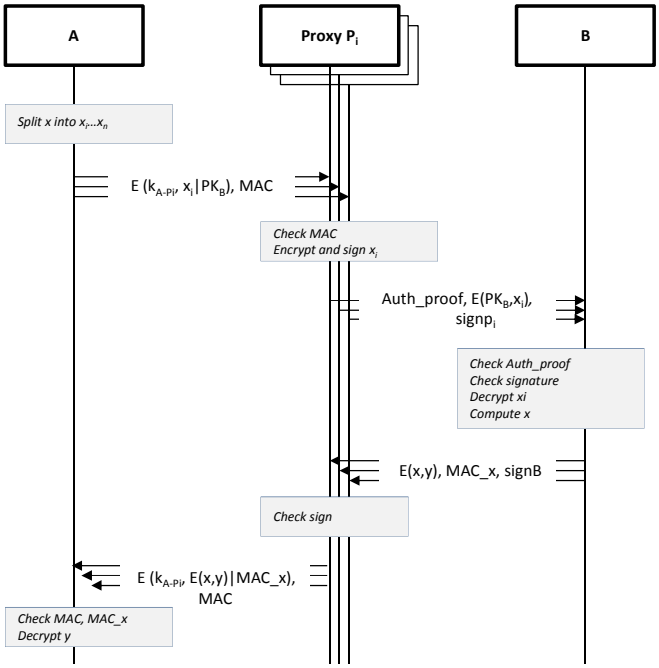


Fig. 12. Collaborative two-pass key transport.

2.3.2. *Collaborative key agreement*

The key agreement process discussed in this subsection involves heavy cryptographic computations at both parties. The most requiring part is the computation of two modular exponentiations, respectively for the generation of the Diffie-Hellman public keys (raise the base g to the power of the secret exponent modulo p) and the setup of the Diffie-Hellman key (raise the peer public value g^x mod p to the power of the secret exponent modulo p). Applying the same collaborative approach as in the

above subsection, we propose to delegate the heavy cryptographic load to less constrained nodes in neighbourhood. The collaborative protocol exchanges are illustrated in figure 13 below, and detailed later in this subsection.

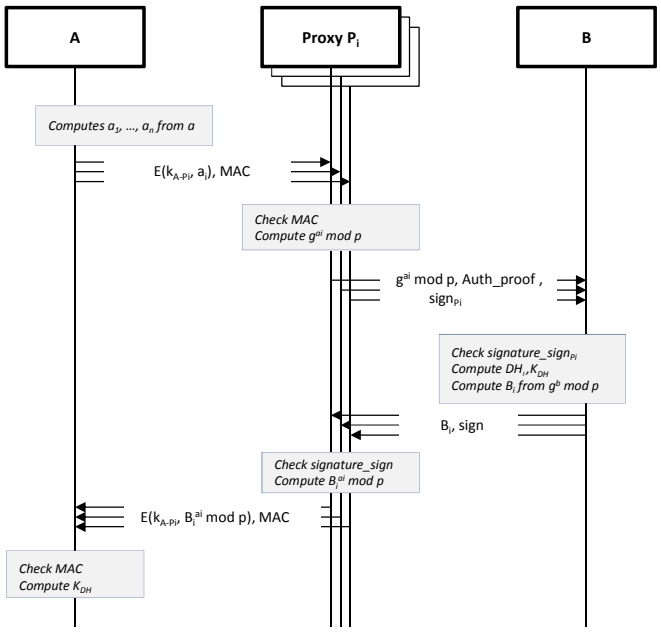


Fig. 13. Collaborative key agreement.

We introduce two techniques to distribute the computations required by the Diffie-Hellman protocol and therefore to enable the key agreement protocol. For each of these techniques, we explain how the source's DH private key is shared among proxies (how A computes the different a_i it gives to each proxy P_i from its secret exponent a), how the server retrieves the source's DH public key from the proxies' $g^{ai} \bmod p$, how the server computes the shares B_i of its own DH public key (each B_i computed by B being sent to the corresponding proxy P_i) and how the proxies use B_i to obtain the K_i shares of the DH session key K_{DH}, eventually used by A to retrieve K_{DH}.

2.3.2.1. Secret exponent integer partition

The integer partition technique is the simplest approach for enabling distributed DH key exchange. The secret exponent a of the source is split into n parts $a_1, \ldots, a_n$ chosen such that:

$$\sum_{1}^{n} a_i = a \bmod p \tag{1}$$

Each a_i is then securely sent to a different proxy P_i. Upon reception of a_i, each proxy P_i computes its part of the initiator's DH public key $g^{a_i} \bmod p$ and delivers it (signed) to the server. The computation of the source's DH public key eventually occurs at the server and amounts to the product of the values received from the proxies, following:

$$\prod_{i=1}^{n} g^{a_i} \bmod p = g^{\sum_{i=1}^{n} a_i} \bmod p = g^{a} \bmod p \tag{2}$$

In turn, the server sends a share B_i of its DH public key to each proxy P_i. In this first simple partition technique, B_i is equal to the server's DH public key for each proxy. The computation by each proxy of the share K_i of the DH session key occurs then as follows:

$$K_i = {B_i}^{a_i} = (g^b \bmod p)^{a_i} = g^{b.a_i} \bmod p \tag{3}$$

Eventually, the computation of the DH session key is made by the source, which obtains K_{DH} as:

$$K_{DH} = \prod_{i=1}^{n} K_i = \prod_{i=1}^{n} g^{b.a_i} \bmod p = g^{b.a} \bmod p \quad (4)$$

According to this expression, the resource-constrained node only spends n-1 modular multiplication operations instead of two modular exponentiation operations, with exponents of considerable length (a and b should have twice the length of the generated secret K_{DH}, as per [58]).

2.3.2.2. Secret exponent threshold distribution

The previous solution is based on reliable multiple hop-by-hop deliveries of secret fragments, each fragment a_i being the ith summand of a modular integer partition of the source's DH private key. The server needs therefore to receive all messages from all proxies in order to be able to reconstitute the source's public key. A single missing message from a proxy makes the information incomplete for the server and may block the protocol exchange.

In order to reinforce the reliability of the proposed distributed scheme, this kind of defective proxy play has been carefully considered in the design of this second proposed approach for key agreement. We have implemented a robust technique that ensures a consistent recovery of the source's DH public key at the server even in case of a proxy misbehaving or unreliability. Note that the redundancy technique introduced above for key transport could not be adapted to a key agreement protocol, which represents a radically different approach where the secret exponent a is never retrieved at B's side.

The enhanced distributed approach we propose is based on the use of a (k, n) threshold scheme, wherein the n proxies obtain a polynomial share of the source secret exponent, k polynomial shares being enough to reconstruct the source secret exponent through the technique of Lagrange polynomial interpolation. This threshold scheme satisfies the two properties that the integer partition solution fails to provide:

1) Recovery: The server can recover the source's public key provided that a sufficient number k of values from proxies are received, without requiring the reception of all of them.

2) Secrecy: Nothing is learned about the secret exponent a even if k-1 shares of it are disclosed. In other words, data delivered to the server through proxies in order to compute the source's public key will not reveal partial information about the secret exponent.

It is worth quickly reminding the operation of the Lagrange polynomial interpolation. Let f be a polynomial function of degree k-1 expressed as: $f(x) = q_0+q_1x+\ldots+ q_{k-1}\, x^{k-1}$ with $q_1, q_2, \ldots, q_{k-1}$ being random, uniform and independent coefficients and $q_0= a$.

From the Lagrange formula, the polynomial f can be retrieved as follows:

$$f(x) = \sum_{i=1}^{k} \left(f(i) \times \prod_{j=1, j\neq i}^{k} \frac{x-j}{i-j} \right) \quad (5)$$

In our threshold key distribution scheme, (5) gives that the secret exponent a can be computed given any subset of k values of $f(x)$:

$$a = f(0) = \sum_{i=1}^{k} \left(f(i) \times \prod_{j=1, j\neq i}^{k} \frac{-j}{i-j} \right) \quad (6)$$

In this threshold distributed approach, the distributed shares a_i of the private exponent a are obtained as $a_i = f(i)$. So, in order to bootstrap the key agreement, the source first calculates the n values $f(1), \ldots, f(n)$ of the polynomial f, with $n > k$, and sends each $f(i)$ to the correspondent proxy P_i. Each proxy computes then its part of the source's DH public key $g^{a_i} \bmod p = g^{f(i)} \bmod p$ and sends it to the server.

Upon the reception of a subset P of k values transmitted by the proxies, the server starts by computing the c_i coefficients as follows:

$$c_i = \prod_{j \in P, j\neq i} \frac{-j}{i-j} \quad (7)$$

Then, B computes the source's DH public key DH_I based on the Lagrange formula:

$$\prod_{i\in P}\left(g^{f(i)}\right)^{c_i} \bmod p = g^{\sum_{i\in P} f(i)\times c_i} \bmod p = g^{f(0)} \bmod p = g^{a} \bmod p \quad (8)$$

In order to prepare the computation of the DH session key at the source side, B starts calculating for each proxy P_i ($i\in P$) the value $B_i = g^{b.c_i} \bmod p$ (c_i being the ith coefficient calculated in the previous phase). P_i is unable to compute the coefficient c_i since it has no knowledge about the subset P of the actually participating proxies. Having received this value, each proxy P_i uses its share $f(i)$ of the source's private exponent to compute $K_i = B_i^{f(i)} = g^{b.c_i.f(i)} \bmod p$. Each proxy delivers then this computed value to the source A.

Upon reception of these k values, the source computes the DH session key K_{DH} as follows:

$$K_{DH} = \prod_{i\in P} g^{b\, f(i)\, c_i} \bmod p = g^{b \sum_{i\in P} f(i)\, c_i} \bmod p = g^{ab} \bmod p \quad (9)$$

By applying the threshold technique to improve the effectiveness of the distributed approach, the source is led to perform more computational operations in the initial phase, in order to calculate the n values of the polynomial that it sends to the n proxies. The cost of the computation can be better estimated if one considers another way of writing $f(x)$, as:

$$f(x) = \left(\cdots\left((q_{k-1}x + q_{k-2}).x + q_{k-3}\right)x + \cdots\right).x + q_0 \quad (10)$$

According to this expression, A performs for each computation of $f(i)$: (k-1) multiplications between a scalar and a large number and (k-1) summations of two large numbers. It is worth noting that k and n are small numbers, smaller than the number of secure relationships that the source is able to maintain. On the other hand, the polynomial coefficients are as large as the DH private key of the source.

2.4. COLLABORATIVE IOT KEY ESTABLISHMENT PROTOCOLS

We consider in this section how our proposed collaborative approach, under its integer partition and threshold distribution embodiments, can be applied to the IoT key establishment protocols that were identified in table 3 of chapter 1.

2.4.1. Modified TLS handshake protocol

As described above, the TLS Handshake Protocol supports two key exchange modes: the one-pass key transport mode and the DH key agreement mode. We modify the protocol exchange considering each of these two modes.

In the following, we assume that the client authentication is performed during the modified TLS handshake protocol. This is in general not the case in the legacy Internet, where human to machine communications take place. Indeed, the server does not require the client certificate and just confirms its identity relying on login/password authentication techniques once the TLS tunnel is established. However, considering the IoT scenarios where machine-to-machine communications are expected, a mutual certificate-based authentication is likely to be required.

2.4.1.1. *Modified TLS handshake in the key transport mode*

The protocol exchange is illustrated in figure 14 below and detailed afterwards. Message exchanges are alike when considering either the threshold secret distribution or the simple secret partition technique. This is because the redundancy scheme is applied at the client before the delivery of the premaster key.

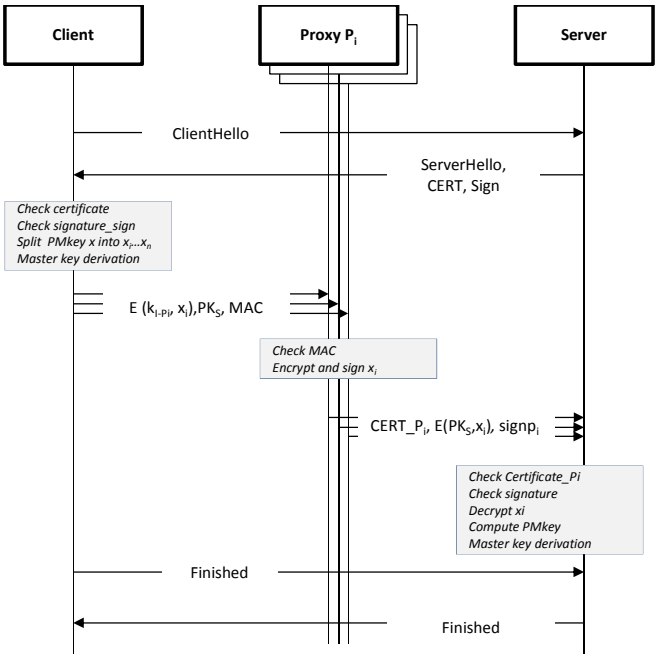


Fig. 14. Distributed TLS handshake (one-pass key transport).

The Hello messages are similar to those of the basic TLS handshake. As described before, both of these messages include random values used as nonces to prevent replay attacks and to compute the session key.

Upon successful connection with the server, the constrained client needs to verify the server certificate (using the Certificate Authority (CA) public key) and signature (using the server public key) and has to securely provide the server with a premaster secret x, used later to compute the shared master key. At this stage, it is worth noting that the verification operations, each performed with an RSA public key, can be supported by the constrained device since they are far less resource-demanding than signature operations involving the use of a private key in RSA cryptosystems (see Table 4 of chapter 1). Delegating these verification operations would be more resource-demanding for the constrained node since it would have first to forward an around 1000 bytes certificate to each proxy, thereby consuming about 29 mJ, for a saving of 3 mJ only.

Once it has verified the legitimacy of the server, the client calls on the proposed cooperative process. It first applies an error redundancy scheme (in case of a threshold secret distribution) to the original premaster key x, splitting it into n parts $x_1,\ldots,x_n$. It then sends each part x_i along with the server public key to the corresponding proxy P_i. At this stage, proxies take in charge the cooperative transmission of the premaster key as described above. The protocol exchange ends with two 'Finished' messages, exchanged between the server and the client, which are computed using the master key and including past exchanges. The 'Finished' messages, as in the TLS basic handshake, are used to ensure that the master key has been correctly recovered at both parties (*mutual key confirmation* property).

2.4.1.2. *Modified TLS handshake in the key agreement mode*

During the key agreement mode, the message exchanges in the threshold secret distribution technique are different from those of the simple integer partition technique. This is because the threshold distributed technique requires more computations at both proxies and server sides during the collaborative key exchange (computation of $g^{b.c_i} \bmod p$ at the server and $g^{b.c_i.f(i)} \bmod p$ at the proxy).

The modified TLS handshake illustrating the two techniques is depicted in the figures 15 and 16 below.

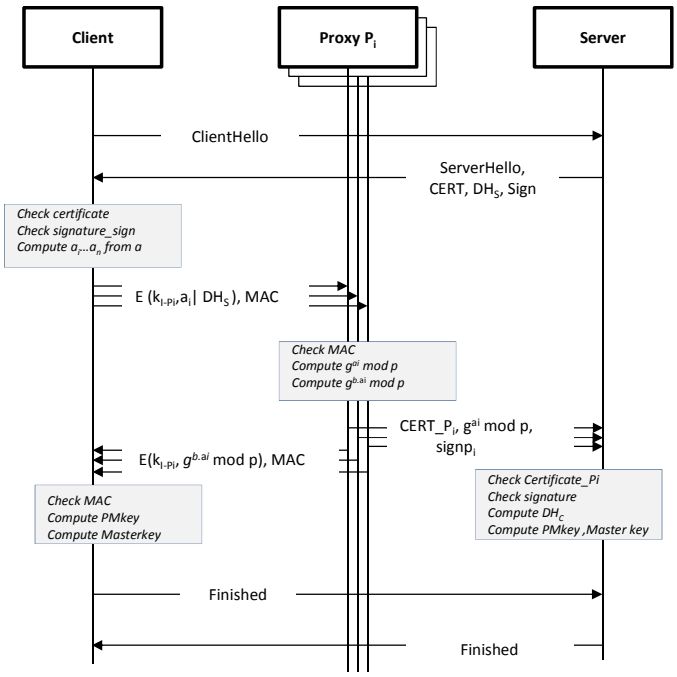


Fig. 15. Distributed TLS handshake: key agreement with simple integer partition technique.

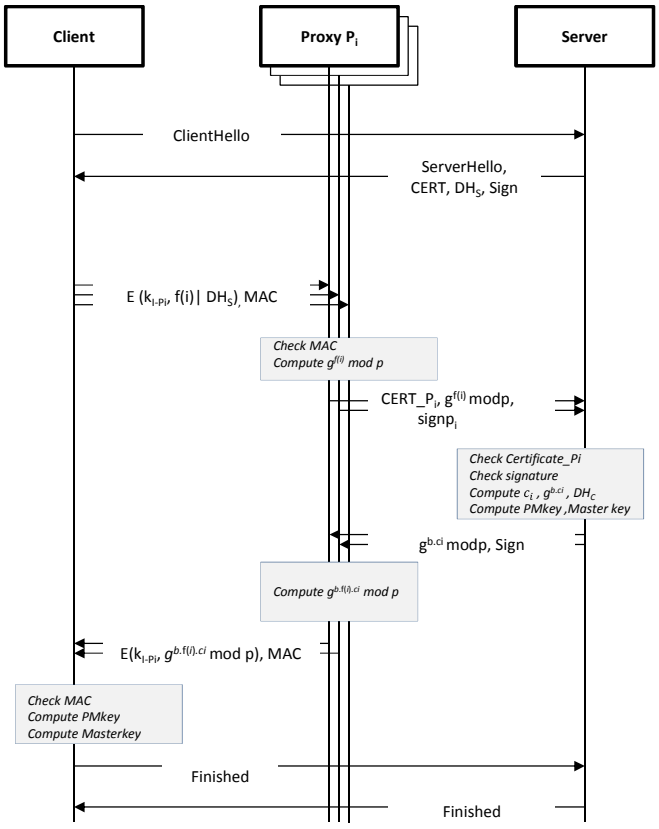


Fig. 16. Distributed TLS handshake: key agreement with threshold secret distribution technique.

During this mode of key exchange, the client offloads the cryptographic operations related to the generation of its DH public key and remains waiting for shares of the DH shared key from proxies in order to be able to eventually derive the master key. Upon receiving replies from proxies, the client performs the (*n* or *k*, depending on the technique) modular multiplications required to recover the DH shared key and becomes then able to compute the master secret at the end of the TLS handshake.

The client and the server then end the protocol handshake by exchanging 'Finished' messages, as in the TLS basic handshake

2.4.2. Modified IKE protocol

The IKE protocol only performs the key agreement mode. The figures below describe the modified protocol exchange obtained by applying the collaborative key agreement with the two proposed techniques.

As in the basic IKE, this modified variant also consists of two phases. During the *IKE_SA_INIT* phase, the two peers perform the Diffie-Hellman key agreement relying on the assistance of proxies as described above and finally derive a master key K_M using both the DH shared key and the nonces (Ni, Nr). During this phase, proxies also provide their certificates to the responder contrary to what happens in the basic protocol exchange. This makes the responder in a position to check the legitimacy of proxies acting on behalf of the initiator and to obtain the reconstitution parameters required to compute DH values. At this stage, proxies' messages are still not authenticated in order to keep authentication process for the second phase, as in the basic IKE.

During the *IKE_AUTH* phase, the initiator delegates the computational load of the signature and verification operations to the proxies in a distributed manner. It first exchanges with the responder encrypted messages using K_M for key confirmation indicating the supported cryptographic algorithms and the proposed traffic selectors (TS_i, TS_r). Then, it triggers the authentication process between the proxies and the server through the message 'AUTH_start' as illustrated in the sequence exchanges below. Once both sides are authenticated, proxies provide the initiator with an 'AUTH_success' message ending the *IKE_AUTH* phase.

Figures 17 and 18 below represent how the proposed approaches with simple integer partition (Fig. 17) and threshold distribution (Fig. 18) are used with the IKE protocol.

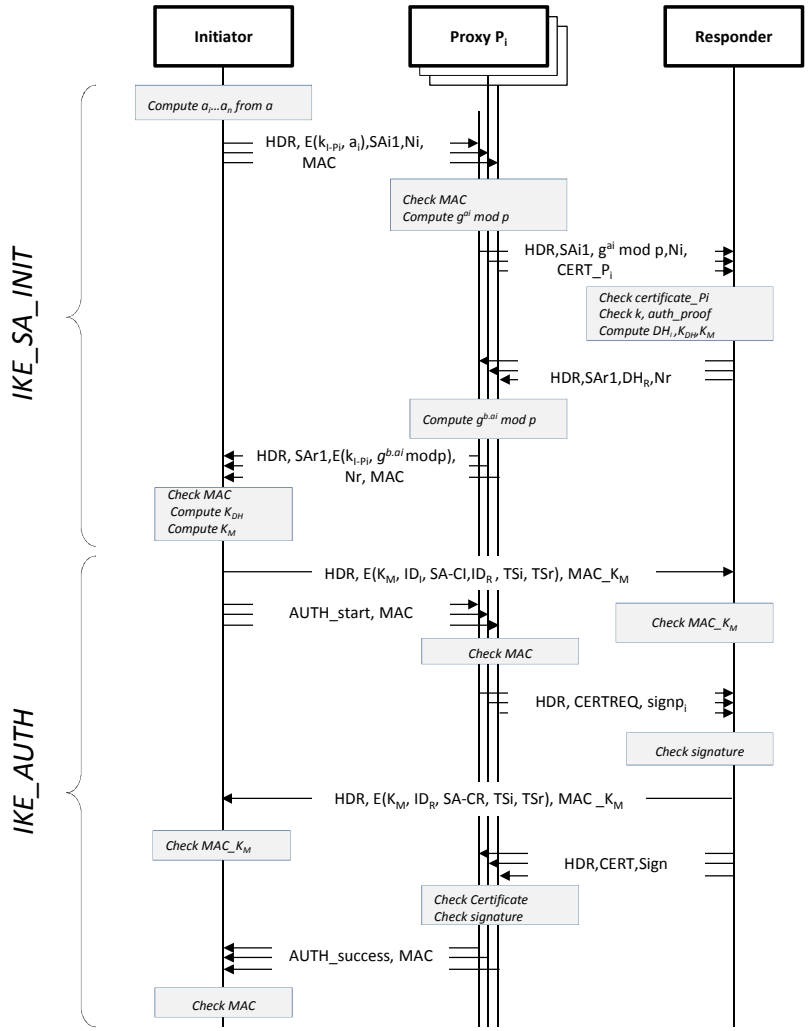


Fig. 17. Distributed IKE: simple integer partition technique.

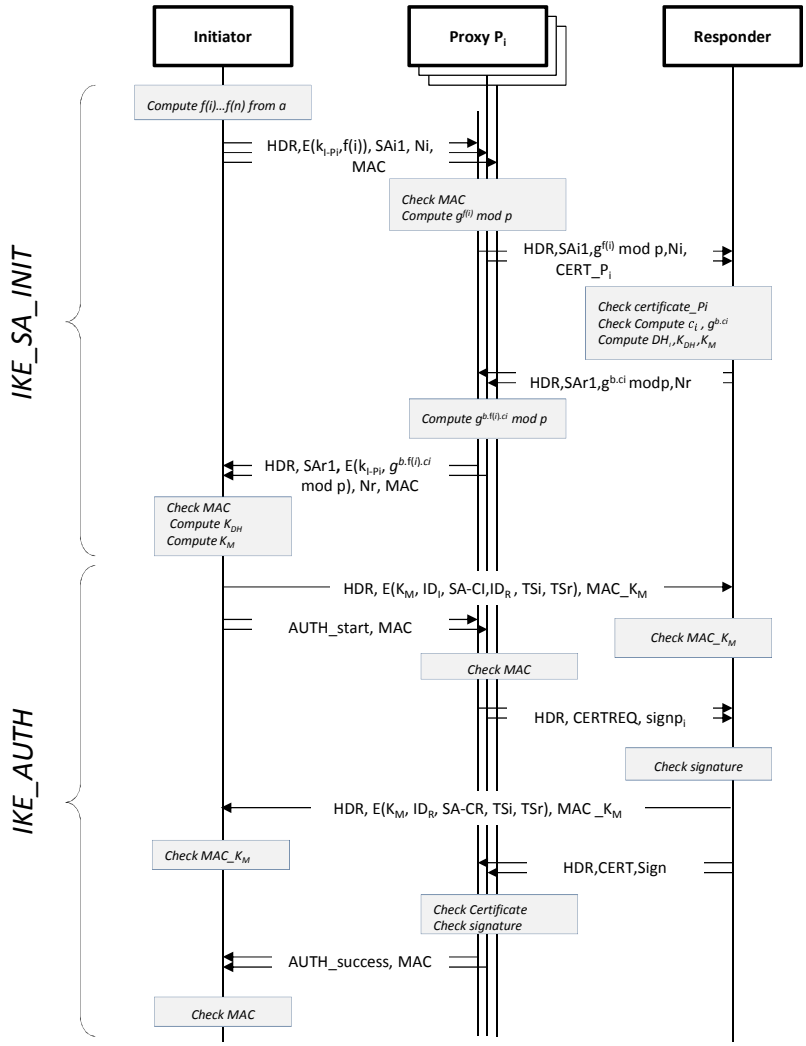


Fig. 18. Distributed IKE: threshold secret distribution technique.

2.4.3. Modified HIP BEX protocol

We illustrate below the modified HIP BEX with the two proposed techniques to distribute the computations required to perform the Diffie-Hellman key agreement.

This lightweight variant keeps the same two first exchanged messages as in the HIP BEX (I1 and I2). Upon receiving the puzzle, the initiator computes the solution and transmits it to the server through proxies within the message $I21_i$. The verification of the responder signature received in the message R1 (around 1.5 mJ) is performed at the initiator since this is less resource consuming than transmitting this 128-bytes signature message to all proxies for verification, which would amount to around 4 mJ. After receiving parts of the initiator secret exponent, proxies provide the server with shares of the initiator DH public key within the message $I22_i$. This message also contains the puzzle solution, the proxy certificate and a signature. Having checked the validity of the solution and the legitimacy of proxies, the server becomes in position to derive the initiator DH public key and the master key. It answers then each participating proxy with a message $R21_i$ similar to message R2 in the BEX, adding a corresponding share ($g^{b.ci}$ mod p) of the DH public value if the threshold key agreement technique is applied. The protocol exchange is finalized by the message $R22_i$ sent from each proxy to the initiator, allowing this latter to compute the master key and check if the result matches with the derived master key at the server.

Figures 19 and 20 below represent how the proposed approaches with simple integer partition (Fig. 19) and threshold distribution (Fig. 20) are used with the HIP BEX protocol.

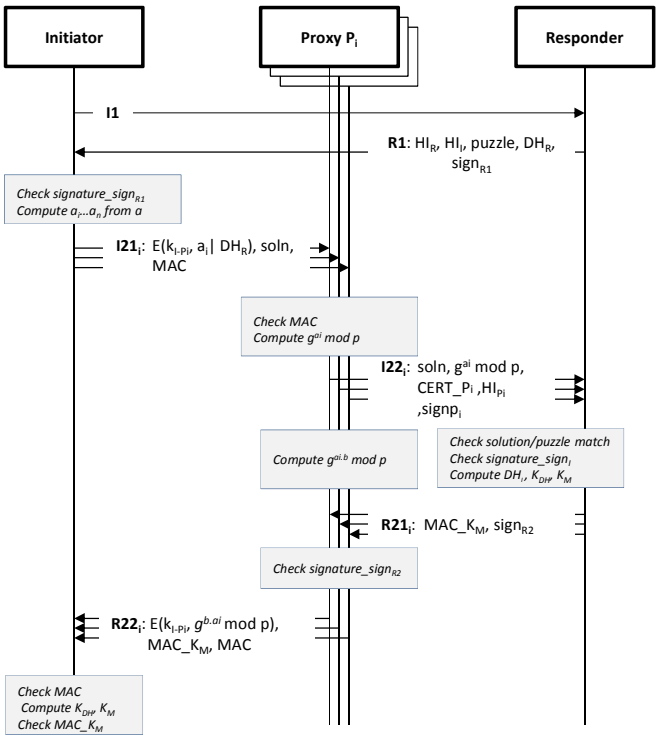


Fig. 19. Distributed HIP BEX: simple integer partition technique.

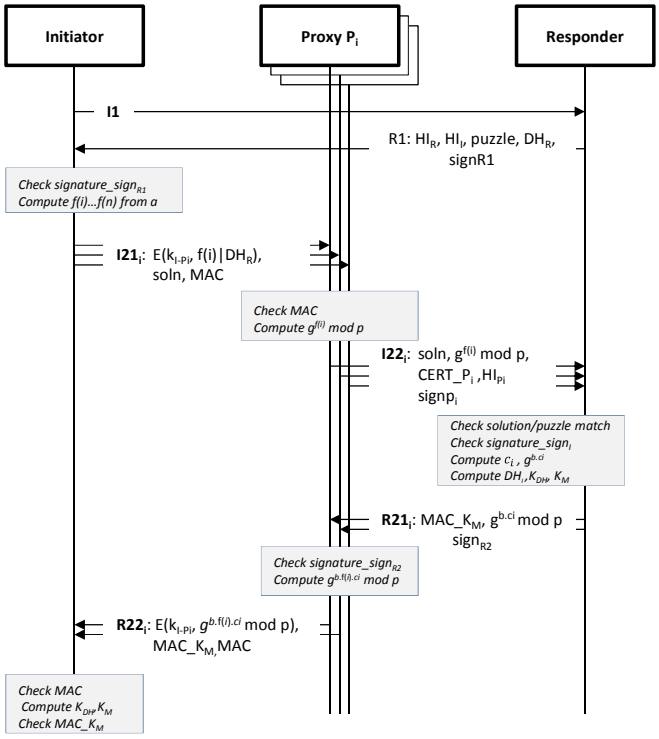


Fig. 20. Distributed HIP BEX: threshold secret distribution technique.

2.5. PERFORMANCE ANALYSIS

As described above, our solution proposes to offload the expensive cryptographic computations to powerful proxies during a key exchange process, hence ensuring significant energy savings at the constrained device. Nevertheless, a communication overhead is imposed due to the message exchanging between the source, the trusted entity T and the proxies.

A performance analysis is therefore required to assess the respective efficiency of the proposed collaborative approaches and compare them with the basic approaches used for the key exchange.

2.5.1. Computational cost

In order to precisely quantify the energy savings at the constrained source node, we have implemented the cryptographic operations it performs in TLS handshake, IKE and HIP BEX protocols, considering both their basic and collaborative approaches. We have evaluated their cryptographic energy costs using Crypto++ library [59]. With respect to error correction, we have chosen to rely on the Reed-Solomon (RS) code [60] in the threshold distributed approach of TLS handshake protocol. In our simulation, we use RS (5, 3) (n = 5, k =3) codes where we generate 2 parity packets for 3 source packets. The computational energy cost of RS code was evaluated using IT++ library [61].

Test programs for individual computational operations were run on an Intel i3 processor and the corresponding number of processor cycles for each was retrieved. In order to be able to induce the energy cost on a resource-constrained device from the number of cycles on a powerful processor, we disabled advanced features on our test processor (hyperthreading, multi-core, variable clock speed). Eventually we were able to consider that the energy cost for a sensor (E_{TelosB}, expressed in Joules) can be derived from the number of cycles measured on the i3 (Ci3), under the following equation:

$$E_{TelosB} = \frac{U_{TelosB}.I_{TelosB}}{N_{TelosB}}.C_{TelosB} = \frac{U_{TelosB}.I_{TelosB}}{N_{TelosB}}.\frac{Register\_size_{i3}}{Register\_size_{TelosB}}.\alpha.C_{i3} \quad (11)$$

Where U, I and N are respectively the voltage, intensity and frequency of TelosB and α is a coefficient representing the richer instructions of the i3 and approximated to 2 in our analysis.

Computational cost results for distributed TLS handshake (representative of a one-pass key transport protocol) and distributed IKE and HIP (representative of key agreement protocols) exchanges are respectively presented in the tables 6, 7, 8 and 9 below.

Table 6: Energy costs of cryptographic operations required by the different evaluated approaches on a TelosB processor for the TLS handshake protocol in key transport mode. (PMK of 48 bytes, AES 128 CBC, HMAC SHA).

	Cryptographic operations	**Energy cost**
Basic approach	verify_CERT+ verify_sign+ RSA_encrypt_x+ RSA_sign_ encrypt_x + compute_Master Key+ compute_Finished+ verify_Finished	2.1 mJ + 1.2 mJ + 1.6 mJ+ 24.43 mJ+ 20.92 µJ+ 267.1µJ + 686.58µJ **=30.30 mJ**
Distr. approach	verify_CERT+ verify_sign+ n*(encrypt_xi+ compute_MAC)+ compute_Master Key+ compute_Finished verify_Finished	2.1 mJ + 1.2 mJ + 5*(2.47µJ+ 16.74µJ)+ 20.92µJ+ 267.1µJ + 573.56µJ **= 4.25 mJ**
Threshold Distr. Approach	verify_CERT+ verify_sign+ encode_reed_solomon+ n*(encrypt_xi+ compute_MAC)+ compute_Master Key+ compute_Finished verify_Finished	2.1 mJ + 1.2 mJ + 350,6 µJ+ 5*(2.47µJ+ 16.74µJ)+ 20.92µJ+ 267.1µJ + 573.56µJ **= 4.6 mJ**

Table 7: Energy costs of cryptographic operations required by the different evaluated approaches on a TelosB processor for the TLS handshake protocol in key agreement mode.

	Cryptographic operations	**Energy cost**
Basic approach	verify_CERT+ verify_sign+ compute_DH_c RSA_sign_ DH_c + compute_K_{DH}+ compute_Master Key+ compute_Finished+ verify_Finished	2.1 mJ + 1.2 mJ+ 58.97 mJ+ 24.48 mJ+ 104.73 mJ+ 20.92 µJ + 267.1 µJ + 686.58 µJ **=192.54 mJ**
Distr. approach	verify_CERT+ verify_sign+ n*(encrypt_a_i+ compute_MAC+ verify_MAC+ decrypt_$g^{b.ai}$modp)+ compute_ mult_$g^{ai.b}$ + compute_Master Key+ compute_Finished+ verify_Finished	2.1 mJ + 1.2 mJ+ 5*(22.25µJ+ 16.74µJ+ 13.57µJ + 19.78 µJ) + 290 µJ+ 20.92 µJ + 267.1µJ + 573.56 µJ + **= 4.81 mJ**
Threshold Distr. Approach	verify_CERT+ verify_sign+ n*(k-1)*(comp_mult_f(i))+ compute_add_f(i))+ n*(encrypt_f(i)+ compute_MAC)+ k*(verify_MAC+ decrypt_$g^{b.f(i).ci}$modp)+ compute_mult_$g^{b.f(i).ci}$ + compute_Master Key+ compute_Finished verify_Finished	2.1 mJ + 1.2 mJ+ 5*2*(0.09 µJ+ 0.05 µJ) + 5* (22.25µJ + 16.74µJ)+ 3*(13.57µJ + 19.78 µJ) + 290 µJ+ 20.92 µJ + 267.1µJ + 573.56 µJ + **= 4.74 mJ**

Table 8: Energy costs of cryptographic operations required by the different evaluated approaches on a TelosB processor for the IKE protocol.

	Cryptographic operations	**Energy cost**
Basic approach	compute_DH_I compute_K_{DH}+ compute_K_M+ compute_sign K_M_encrypt_msg3+ compute_MAC_K_M+ verify_MAC_K_M + K_M_decrypt_msg4+ verify_CERT+ verify_sign	58.97 mJ + 104.73 mJ + 16.74 µJ + 24.39 mJ + 205.25 µJ+ 142.31 µJ+ 138.12 µJ+ 200.31 µJ+ 2.1 mJ + 1.22 mJ + **=192.11 mJ**
Distr. approach	n*(encrypt_a_i+ compute_MAC+ verify_MAC+ decrypt_$g^{b.ai}$modp)+ compute_ mult_$g^{ai.b}$ + compute_ K_M + K_M_encrypt_msg3'+ compute_MAC_K_M + verify_MAC_K_M + K_M_decrypt_msg4'+ n*(compute_MAC+ verify_MAC)	5*(2.47µJ+ 10.46µJ + 23.02µJ + 19.78 µJ) + 290 µJ+ 16.74µJ 29.67µJ 23.02 µJ 18.83 µJ 24.73 µJ 5*(2.1 µJ + 2.1 µJ) **= 702.64 µJ**
Threshold Distr. Approach	n*(k-1)*(comp_mult_f(i))+ compute_add_f(i))+ n*(encrypt_f(i)+ compute_MAC) + k*(verify_MAC+ decrypt_ $g^{b.f(i).ci}$modp)+ compute_ mult_$g^{b.f(i).ci}$ + compute_ K_M + K_M_encrypt_msg3'+ compute_MAC_K_M + verify_MAC_K_M + K_M_decrypt_msg4'+ k*(compute_MAC+ verify_MAC)	5*2*(0.09 µJ+ 0.05 µJ) + 5*(2.47µJ+ 10.46µJ)+ 3*(23.02µJ + 19.78 µJ) + 290 µJ+ 16.74µJ 29.67µJ 23.02 µJ 18.83 µJ 24.73 µJ 3*(2.1 µJ + 2.1 µJ) **= 610.04 µJ**

Table 9: Energy costs of cryptographic operations required by the different evaluated approaches on a TelosB processor for the HIP BEX protocol.

	Cryptographic operations	Energy cost
Basic Approach	verify_signR1+ compute_DH_I+ compute_soln+ compute_signI2+ compute_K_{DH}+ compute_K_M + verify_signR2+ verify_MAC_K_M	1.24 mJ + 58.97 mJ + 135.6 µJ+ 24.55 mJ + 104.73 mJ + 16.74 µJ+ 1.24 mJ + 2.1 µJ = **190.88 mJ**
Integer Partition	verify_signR1+ compute_soln+ n*(encrypt_a_i + compute_MAC + verify_MAC+ decrypt_$g^{b.ai}$modp) + compute_mult_g^{aib} + compute_K_M+ verify_MAC_K_M	1.24 mJ + 135.6 µJ+ 5*(22.25µJ+ 18.83µJ+ 16.74µJ + 19.87 µJ) + 290 µJ + 16.74 µJ+ 2.1 µJ =**2.07 mJ**
Threshold Distr. Approach	verify_signR1+ compute_soln+ n*(k-1)*(comp_mult_f(i)) + compute_add_f(i)) + n*(encrypt_f(i)+ compute_MAC) + k*(verify_MAC+ decrypt_ $g^{b.f(i).ci}$modp)+ compute_ mult_$g^{b.f(i).ci}$ + compute_ K_M + verify_MAC_K_M	1.24 mJ + 135.6 µJ+ 5*2*(0.09 µJ+ 0.05 µJ) + 5*(22.25µJ+ 18.83µJ) + 3*(16.74µJ+ 19.87 µJ) + 290 µJ + 16.74 µJ+ 2.1 µJ =**2 mJ**

2.5.2. Communication cost

In this subsection we assess the communication energy costs of the proposed distributed approaches at the constrained initiator. These costs are made of the costs of transmission, reception and listening. The energy consumption of a node in listening mode can be equivalent to its consumption in reception mode since the transceiver remains active in both modes (see Table 10). Nevertheless, most of existing works do not consider the listening mode in their communication cost evaluations

Authors in [62] assess the energy cost of cryptographic algorithms is WSNs nodes and reveal the impact of listening on the total energy cost. However they did not consider this element in their estimates. Reference [63] includes the listening cost to estimate the energy cost of ECDH-ECDSA and Kerberos protocols on TelosB and MICAz sensors and insists on its importance comparing results with a prior work that estimates communication cost considering only transmission and reception costs. This comparison shows an energy overhead of 45% when the listening cost is taken into account.

Table 10: Power consumption of TelosB at 4 MHz with a transmit power of -5 dBm (from [63]).

	TelosB platform
Transmit	54 mW
Receive	61 mW
Listen	60 mW

We use the power consumptions presented in the Table 10 as an energy model of the different operating modes (transmit, receive and listen) for the TelosB platform [63]. As reported in [63] we consider an effective data rate of 75 kbps instead of a 250 kbps claimed one. This important decrease of the data rate is discussed in [64]. In a nutshell, both the presence of headers and footers and the use

of acknowledgments combine with the expected nominal–effective decrease to further diminish the rate available for application data.

From the previous exchange descriptions, we obtain in the table 11 below the number of exchanged bytes by the source node in TLS handshake, IKE and HIP BEX protocols, considering both the basic exchange and the distributed approaches.

Table 11: Sent and received bytes in the TLS handshake, IKE and HIP BEX protocols.

	TLS handshake protocol (key transport mode)			**TLS handshake protocol (key agreement mode)**			**Internet key exchange protocol**			**HIP BEX protocol**		
	Basic approach	Distributed approach	Threshold distributed approach	Basic approach	Integer partition approach	Threshold distributed approach	Basic approach	Integer partition approach	Threshold distributed approach	Basic approach	Integer partition approach	Threshold distributed approach
Sent (bytes)	**2367**	**2095**	**2095**	**2495**	**2863**	**2863**	**1568**	**968**	**932**	**468**	**952**	**952**
Recv (bytes)	**4610**	**3484**	**3484**	**4994**	**4502**	**4354**	**1542**	**1496**	**1236**	**608**	**1140**	**972**

We consider that the constrained node is listening during a delay corresponding to the latency of communications (T_x, R_x) and packets propagation (Δ) as well as the processing of packets (Proc) at the proxies and the server. We estimate below the listening durations required by the constrained node in the considered approaches:

$$\Delta t_{Listen} = \underbrace{Proc(P_i) + \underbrace{T(P_i) + \Delta(P_i \rightarrow R) + Proc(R) + \Delta(R \rightarrow P_i) + R(P_{i)}}_{Basic\ approach\ listening\ time} + Proc(P_i)}_{Distributed\ approach\ listening\ time}$$

Assuming that the server is an unconstrained node while proxies are 10 times less constrained than the server (and thus have a 10-time greater processing time), the listening durations for the different keying approaches are presented in the table 12 below.

Table 12: Listening durations (in ms) in the four considered key establishment protocols (basic & distributed).

TLS handshake protocol (key transport mode)		**TLS handshake protocol (key agreement mode)**		**Internet key exchange protocol**		**HIP BEX protocol**	
Basic approach	*Distributed approaches*	*Basic approach*	*Distributed approaches*	*Basic approach*	*Distributed approaches*	*Basic approach*	*Distributed approaches*
401	**411**	**404**	**444**	**404**	**446**	**405**	**445**

We also assume that the proxy is one hop far from the constrained node and that a 200 ms propagation delay is required to route packets from the source to the server. Finally, the energy costs induced by communications in both basic approach and distributed approaches is shown in tables 13, 14, 15 and 16 below.

Table 13: Communication Energy costs on a TelosB processor for the TLS handshake protocol in key transport mode.

	Basic Approach	**Distributed Approach**	**Threshold Distributed Approach**
Transmit cost	13.63 mJ	12.06 mJ	12.06 mJ
Receive cost	29.87 mJ	22.57 mJ	22.57 mJ
Listen cost	24.06 mJ	24.66 mJ	24.66 mJ
Energy cost	67.56 mJ	59.29 mJ	59.29 mJ

Table 14: Communication Energy costs on a TelosB processor for the TLS handshake protocol in key agreement mode.

	Basic Approach	**Integer partition Approach**	**Threshold Distributed Approach**
Transmit cost	14.37 mJ	16.49 mJ	16.49 mJ
Receive cost	32.36 mJ	29.17 mJ	28.21 mJ
Listen cost	24.24 mJ	26.64 mJ	26.64 mJ
Energy cost	70.97mJ	72.3 mJ	71.34 mJ

Table 15: Communication Energy costs on a TelosB processor for the IKE handshake protocol.

	Basic Approach	**Integer partition Approach**	**Threshold Distributed Approach**
Transmit cost	9.03 mJ	5.57 mJ	5.36 mJ
Receive cost	10 mJ	9.69 mJ	8 mJ
Listen cost	24.24 mJ	26.76 mJ	26.76 mJ
Energy cost	43.27 mJ	42.02 mJ	40.12 mJ

Table 16: Communication Energy costs on a TelosB processor for the HIP BEX protocol.

	Basic Approach	**Integer partition Approach**	**Threshold Distributed Approach**
Transmit cost	2.7 mJ	5.48 mJ	5.48 mJ
Receive cost	3.93 mJ	7.38 mJ	6.3 mJ
Listen cost	24.3 mJ	26.7 mJ	26.7 mJ
Energy cost	30.93 mJ	39.56 mJ	38.48 mJ

2.5.3. Total energy cost

Synthesizing the computation and communication costs, we provide the total energy costs of the two examples of key exchange protocols considering the basic and collaborative approaches in figure 21 and table 17 below.

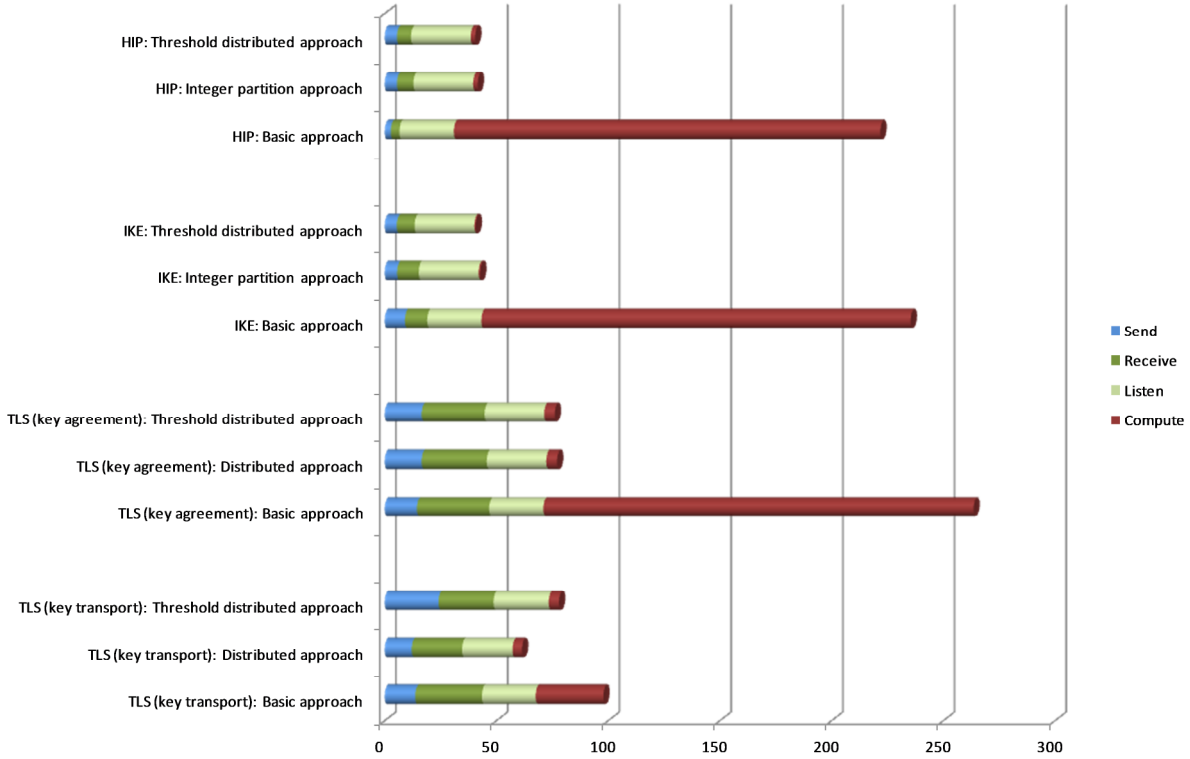


Fig. 21. Overall energy consumption on a TelosB in the four considered key establishment protocols for basic & distributed approaches (considering the basic and resilient modes for the distributed approaches).

Table 17: Compared total (computations + communications) energy costs on a TelosB processor for the retained IoT key establishment protocols, featuring for each protocol the basic (unchanged) approach, the default collaborative approach and the resilient collaborative approach.

	TLS handshake protocol (key transport mode)			**TLS handshake protocol (key agreement mode)**			**Internet key exchange protocol**			**HIP BEX protocol**		
	Basic approach	Distributed approach	Threshold distributed approach	Basic approach	Integer partition approach	Threshold distributed approach	Basic approach	Integer partition approach	Threshold distributed approach	Basic approach	Integer partition approach	Threshold distributed approach
Comp.	30.30	4.25	4.6	192.54	4.81	4.74	192.11	0.702	0.610	190.88	2.07	2
Comm.	67.56	59.29	59.29	70.97	72.3	71.34	43.27	42.02	40.12	30.93	39.56	38.48
Total energy cost (mJ)	**97.86**	**63.54**	**63.89**	**263.51**	**77.11**	**76.08**	**235.38**	**42.72**	**40.73**	**221.81**	**41.63**	**40.48**

As shown in figure 21 and table 17, the computed costs confirm the efficiency of the cooperative scheme we propose. The most significant energy savings concern the key agreement mode. They amount to 75% of what is consumed in the key agreement mode of TLS handshake and 80% of what is consumed in IKE and HIP BEX protocols. Concerning the key transport of TLS handshake, the constrained node saves around 35 % of its energy, as compared with what is spent during the basic exchange. These results were expected since delegating the computation of DH modular exponentiations (in the key agreement mode) leads to more energy savings at the constrained device than offloading signature and encryption operations in the key transport mode.

Energy savings can be increased by reducing the duration of listening mode. Using LPL (Low Power Listening) protocols [65], the source node can be temporarily put into a sleep mode when waiting for the protocol to run between proxies and server. These saving can be especially important for the key agreement protocols, where the listening communication cost amounts to more than 50% of the overall energy consumption.

The results also show that the energy costs of the threshold distributed approaches of the key agreement mode in the three studied protocols are slightly less small than those of the simple distributed approaches; contrary to what may have been expected if one had only considered the additional cost of the generation of the polynomial shares. This generation overhead certainly makes the secret distribution more complex, but meanwhile it reduces the energy cost of messages processing at the source node, which receives and deciphers k packets instead of n .These k packets contain shares of DH session key sent from proxies at the end of the protocol exchange to make it possible for the source node to set up the master key.

Concerning the key transport mode of TLS handshake, the overhead introduced by the addition of redundant parity packets in the threshold distributed approach slightly increases the energy cost of the protocol exchange. On the other hand, the constrained source is not expected to process packets received from proxies so that the introduced overhead is not compensated as in the key agreement mode.

In a nutshell, simulation results prove the viability of the proposed distributed approaches in the studied context of IoT keying, which involves highly resource-constrained nodes such as the TelosB sensor platform. Providing almost equivalent energy costs compared to the simple distributed approach, the threshold distributed approach introduces additional recovery and secrecy properties, both essential for a collaborative protocol.

After proving the efficiency of the proposed collaborative variants of TLS handshake, IKE and HIP BEX, a formal security analysis of these approaches is required in order to prove their overall effectiveness as key establishment security protocols.

2.6. FORMAL VALIDATION WITH AVISPA

A formal security analysis was carried out using the AVISPA [66] tool in order to prove the fulfillment of the desired security goals of the proposed collaborative keying schemes. AVISPA (Automated Validation of Internet Security Protocol and Applications) is a push button security protocol analyser based on formal methods, performing analytical rules to illustrate whether the candidate protocol is safe or not. If a vulnerability is detected, verification results revolve the attack trace, showing at which step and under which conditions an attack was made possible. The tool implements the Dolev-Yao intruder model [67] able to eavesdrop, intercept messages, insert bogus data, or modify traffic passing through. AVISPA incorporates four different automatic protocol analysis techniques for protocol falsification on-the-fly model-checker (OFMC), constraint-logic based attack searcher (CL-AtSe), SAT-based model checker (SATMC), and tree automata based on automatic approximations for the analysis of security protocols (TA4SP) and provides a large library of well-known Internet security protocols.

The first step of the protocol verification consists in modeling it using HLPSL formal language of AVISPA. The specification language HLPSL is used to describe the security protocol as sequences of exchanged messages between different parties and to express desired properties and security goals. Actors interacting in the exchange are modeled as the roles including their message exchanges with each other. After that, a session is created by binding the roles altogether, describing message exchanges in a normal run of the given protocol. Other sessions are then specified, with the difference that they include an active intruder in between the different actors, specifying its optional knowledge of keys known to legitimate entities. Modeling the intruder activity is used to interactively find and build attacks over the present protocol. Finally a global environment is created including multiple parallel sessions simultaneously. The HLPSL specification is later translated into an Intermediate Format specification providing a low-level description of the protocol and given as an input to the four automatic analysis back-ends of the AVISPA tool. Then the verification of the security properties of the protocol, namely authentication, integrity, anti-replay and secrecy, starts. If a specified security

property is violated, the back-ends return a trace explaining the sequence of actions that gave rise to the attack and exhibit which goal was violated.

We specified first the actions of each participant in a module, which is called a basic role; the role of the constrained client in modified TLS handshake protocol, for example, is modeled as follows:

```
role client(A, B, Pi, T : agent,
            Kat, Kapi, Kpit: symmetric_key,
           Kb,Ks: public_key, %ks is the public key of a Trusted Third Party
            SND_BA, RCV_BA, SND_PiA, RCV_PiA,          SND_TA, RCV_TA: channel (dy))
played_by A
```

The declared variables above represent the initial knowledge of the client. The RCV and SND parameters indicate the channels upon which the participant playing "role client" will communicate with other roles. Here the client A communicates with the server B and P_i, both sending and receiving packets.

In the same form, the role proxy in the modified HIP BEX is modeled as follows:

```
role proxy (A, B, Pi, T : agent,
        Hash,Soln:hash_func,
         Kat,Kapi,Kpit:symmetric_key,
         HI_T,HI_B : public_key,
         G:nat,
         SND_TPi, RCV_TPi, SND_APi, RCV_APi,SND_BPi, RCV_BPi: channel (dy))
played_by Pi
```

Each role consists of a sequence of states illustrating all of its exchanges with other parties involved in the protocol. The state below describes the exchange between the initiator and the proxy in the modified HIP BEX. Having received the list of proxies participating in the key exchange from the trusted entity T, the client A sends a message to the proxy P_i containing the solution of the puzzle, the server DH public value, the server identifier and a part of the secret exponent a_i.

```
 State = 8
     /\ RCV_TA({Pi}_Kat)
     =|>
     State' := 10
     /\ Xi' := new()
     /\ SND_PiA({Soln(Puzzle). {exp(G,Y).HI_B}_inv(HI_B). ai'}_Kapi)
```

After defining basic roles, we defined composed roles which describe the whole session by the execution of all basic roles simultaneously.

```
role session( )
Def=
local
composition
                        client()
```

```
/\ trustparty()
/\ proxy()
/\ server()
```

Finally, a top-level role called “environment” was defined including the intruder activity trying to play some roles as a legitimate user. The environment role in the modified IKE protocol is modeled as follows:

```
role environment()
def=
  const
      server_proxy,proxy_server: protocol_id ,
      hash_, mult_: hash_func,
      a,b,pi,t  : agent,
      kt,kb,ks,ki : public_key,
      g,ni,nr : nat,
      kat,kapi,kpit,kipi,kai,kti:symmetric_key
  intruder_knowledge = { a, b, pi, t, g, ni, nr, hash_,mult_, kt,kb,ks,ki, kipi,kai,kti, inv(ki), {i.ki}_(inv(ks))}
  composition
     session(a,b,pi,t,hash_,mult_, kat,kapi,kpit,g, kt,kb,ks)
   /\ session(a,b,i,t,hash_,mult_ ,kat,kai,kti,g, kt,kb,ks)
   /\ session(a,i,pi,t,hash_,mult_,kat,kapi,kpit,g, kt,ki,ks)
   /\ session(i,b,pi,t,hash_,mult_, kti,kipi,kpit,g, kt,kb,ks)
end role
```

In the above extract, one can notice that the modeled intruder may have had its public key (i.ki) signed by the same certificate authority that authenticates B, as represented by its knowledge of an {i.ki}_(inv(ks)) statement. Another noticeable point is the variation of the roles that the intruder i may assume in the protocol test, as shown in the last three lines: i is successively described as being able to act as P_i, A and B.

The security goals were finally specified in a “goal” section asserting that the secrecy should be achieved for the final master key between the client A and the server B, and for the Lamport private key (we model Lamport signatures at the proxy) between the trusted party T and each proxy.

The secrecy of a parameter was also declared before, in the “role” section of the agent who has generated it. For example, after the generation of the Lamport key material in the role of the trusted party, we have further described the transition (exchanges) with the following secret facts:

```
/\ Kpi' := new() % material key generation
/\ secret (inv(Kpi'),k,{T,Pi})
```

This means that the trusted party T declares that the generated Lamport private key is kept secret between T and P_i only and that this security objective is to be referred to as ‘k’.

In a second part of the “goal” section, we asserted that authentication should be verified between each proxy and the server in order to prove that the node is legitimate and authorized to act on behalf of the constrained node and that the proxy communicates with the desired entity.

Goal facts related to the mutual authentication between the proxy and the server are stated at the role proxy and role server sections. The goal fact “witness” is used by the role to be authenticated in

order to express that he wants to be the peer of the other role and will prove later its legitimacy. The goal fact "request" preceded by an accompanying witness is used by the authenticating role releasing in the transition after which the authentication is verified and is considered successful.

In our modified protocols, we have used witness and request facts for the mutual authentication between the proxy and the server. Example depicted below concerns the modified IKE:

• The proxy authenticates the server on the value of Ni (because the server implicitly sends back the received fresh nonce Ni signed with its private key). Actually, the server signs a message encrypted with the master key which was computed using the nonce Ni. This translates as:

```
/\ witness(B,Pi,proxy_server,Ni')  (at the role server)
/\ request(Pi,B,proxy_server,Ni)   (at the role proxy)
```

• The server authenticates the proxy on the value of Nr (because the proxy implicitly sends back the received fresh nonce Nr signed with its Lamport private key). This translates as:

```
/\ witness(Pi,B,server_proxy,Nr')  (at the role proxy)
/\ request(B,Pi,server_proxy,Nr)   (at the role server)
```

Eventually, these three goals (secrecy of (KM, k) and mutual authentication between proxy and server) translate to:

```
goal
 secrecy_of k,km
 authentication_on server_proxy
 authentication_on proxy_server
```

Subsequently, we checked the correctness of the implemented HLPSL codes and of the protocol state machines by the use of the protocol animation tool called SPAN [68].

Finally, the security of the protocols was evaluated by executing the four AVISPA back ends (OFMC, SATMC, CL-AtSe and TA4SP) against our defined intended security goals. Peer authentication, secrecy, message integrity, delivery proof, identity proof and replay protection were evaluated. AVISPA tool produced a formal report as an output indicating that the protocol is "SAFE" against OFMC, CL-AtSe, and SATMC and "INCONCLUSIVE" against TA4SP database. No vulnerabilities were detected: according to the tool, it is not possible for an intruder to violate a security requirement and alter the successful protocols run, based on the specified security goals and the described assumptions. The output is provided as follows:

```
AVISPA Tool Summary
OFMC    : SAFE
CL-AtSe : SAFE
SATMC   : SAFE
TA4SP   : INCONCLUSIVE
```

2.7. CONCLUSION

This chapter presents a novel collaborative approach for key establishment in the context of the IoT, by which a resource-constrained device delegates its expensive computational load to assisting nodes, on a distributed and cooperative basis. In order to enable this collaborative behaviour, two distributed techniques have been proposed and carefully designed for both the key transport and key agreement modes. These techniques have been applied to redesign retained key establishment standards for the

IoT, as identified in chapter 1, namely TLS handshake, Internet Key Exchange and HIP Base EXchange protocols.

The cooperative variants of these protocols have then been assessed and compared to the legacy key establishment protocols they base on, from the points of view of cryptographic and communication costs. Simulation results show that our proxy-based scheme significantly increases the energy savings at the constrained device compared with existing standards.

A formal security analysis performed using AVISPA tool has validated the security of the modified exchange protocols against external attackers attempting to violate the major properties related to a communication security protocol, that is authentication, confidentiality, freshness and data integrity.

However, the obvious benefits of our collaborative approach should not hide the new threats they introduce that AVISPA tool is unable to reason about. The IoT is also characterized by the fact that it interconnects within a single infrastructure a wide variety of entities, some of which being expected to become compromised and act maliciously over time. When nodes rely on each other to achieve a common goal, more points of failure arise that may deter the efficient service fulfilment. A legitimate proxy can act selfishly and refuse to participate to the collaborative key exchange process in order to save its energy resources and maximize its own performance. Or it can act maliciously and impair the collaborative process with the goal of damaging the whole system. These types of threats, introduced by collaborative aspects, are known as internal attacks. Conventional cryptographic mechanisms such as signature and encryption can provide confidentiality, integrity and node authentication for exchanged messages and protect the system from external attacks; however, they fail to deal with insider attackers since the misbehaving proxy is often certified by a trusted authority to be a legitimate entity.

As explained throughout this chapter, this kind of "unfair" proxy play has been carefully considered in the design of our collaborative approach. Threshold techniques have been implemented during the key exchange for ensuring a consistent recovery of the secret key in case of a proxy unavailability or misbehaviour. Nevertheless, further security measures have to be considered in order to identify malicious participants through an analysis of their behaviour inside the cooperative group. This identification process is essential to isolate untrustworthy elements and refine future proxy selections.

In the literature, collaboration between nodes has been proposed for enabling various networking services, with the objective to improve the communications between any two nodes in a networked infrastructure. Accordingly, behaviour analysis systems were designed that aimed at securing the proposed collaborative schemes. We conducted a review of these systems in order to assess the different forms that collaboration management could take. Especially, our objective was to analyse whether any existing behaviour analysis systems could fulfil the specific requirements of our collaborative key establishment schemes. This will make the subject of the next chapter.

Chapter 3: COLLABORATIVE SERVICES AND THEIR SECURITY-RELATED WORK

3.1. INTRODUCTION

Recently, we have witnessed the emergence of collaboration between nodes in wireless communication systems to accomplish jointly a specific task or to maximize the overall system performance. Collaboration has gained momentum with the advent of new communication schemes introducing unattended wireless topologies, mostly made of resource-constrained nodes, in which radio spectrum therefore ceased to be the only resource worthy of optimisation. Collaborative techniques are introduced to improve the performance of wireless topologies in many respects, for example by increasing the coverage, enhancing the security or saving bandwidth and energy resources.

Along the same lines of our solution, other collaborative services have been proposed in the literature. Among these, we chose to focus on collaborative networking services, which we define as featuring functions that improve the communication abilities of any two networked nodes. Radio connectivity, end-to-end routing, establishment of secured channels fit within this definition. On the other hand, it excludes both orchestrated applicative services and services that essentially rely on assigning different roles to the connected entities, such as aggregation or backup.

This chapter starts in section 3.2 by presenting the different networking services for which collaborative approaches have been proposed in the literature. Next, we review in section 3.3 the security measures that are proposed to counter internal attacks that can be launched inside a collaborative group. By assessing existing behaviour analysis mechanisms, we build in section 3.4 a synthesis of the best practices to use as part of a generic trust management system. We conclude this chapter in section 3.5.

3.2. COLLABORATIVE NETWORKING SERVICES IN WIRELESS COMMUNICATIONS

In this section, we survey existing collaborative networking services in wireless communications. The considered collaborative processes in our comprehensive approach include routing, security and radio services.

3.2.1. Collaborative routing services

In a WSN, the main application of sensor nodes is to collect and report events to a sink node. Collected data delivery is provided through multi-hop communications, since direct communications between sources and the sink node could be not feasible for sensor nodes, because of their constraints in terms of transmission range or limited energy. Hence, collaborative routing schemes able to support distant communication with a sink node prove out to be a necessity in WSNs. Intermediate sensor nodes collaborate to forward packets between the source and the sink node. If clustering is applied, dedicated nodes are deployed in the sensor network to support the transmission burden from sensors to the sink node. The network is then divided into a group of clusters.

A cluster head with richer resources capabilities receives collected data from sensor nodes within its own cluster, and delivers them to the sink node. This hierarchical collaboration between sensor nodes and cluster heads to route data has been proposed to achieve energy efficiency in WSNs. Collaboration arises also as an essential requirement in Mobile Ad-hoc NETworks (MANETs) routing. The lack of a fixed infrastructure in a MANET leads to decentralized communications between nodes, therefore causing the routing activities to be carried out by participants. A mobile node is seen as a communicating node as well as a relay node that collaborates with other nodes to forward and route messages from a source to a destination.

Collaboration between nodes for routing and packet forwarding is seen as an inherent behaviour [69] as compared with other networking services. By essence, routing involves intermediate nodes between the sender and the recipient of a packet that are in charge of forwarding the sent packet until it has reached its final destination. Routing also involves dedicated control-plane messaging between nodes allowing them to build awareness of their neighbours' own routing capabilities, in order to determine the optimal route to send a packet. Existing routing protocols such as AODV [70] and DSR [71] assume that all the nodes that form the wireless network have to cooperate and are inclined to act as assisting nodes in a routing process by forwarding packets of other nodes in the network.

3.2.2. Collaborative security services

Recent years have witnessed an increased interest in the concept of collaboration as a technique to apply for enabling security services. Collaboration has first been suggested by cryptographers to deal with secret sharing. The concept of secret sharing was introduced in 1979 by Shamir [72] and Blakley [73] based respectively on Lagrange interpolating polynomial and Linear projective geometry, as a solution to cryptographic keys management. The basic idea consists in splitting a dealer's secret into multiple shares and distributing the result among a set of participants. Then a subset of these participants belonging to the access structure can collaborate to combine their shares and recover the secret when needed. Such schemes have also been referred to as (k, n) threshold secret sharing schemes since the secret is retrieved only if at least k from n participants ($n > k$) cooperate to combine their shares. Secret sharing schemes were proposed to protect and control the access to any important information in the network by distributing it over different locations, thereby imposing an attacker to have access to these multiple locations in order to learn about the information [74].

Another security service in which collaboration is required is signature delegation, also known as proxy signature, whose concept was put forward in 1996 by Mambo et al [75]. The primitive of proxy signature allows a proxy to sign a message on behalf of an original signer. This latter delegates its signing authority to a designated proxy, mandated to act on its behalf. However, relying on a single proxy node makes the security of the proposed scheme dependent on the reliability of the proxy signer and impractical.

In order to share signing responsibilities, the concept of proxy signature was therefore extended to delegate signing rights to a group of participants [76]. Each participating proxy initially receives a partial proxy signing key from the original signer. Then, proxies collaborate to generate a valid proxy signing key, required to act on behalf of the original node. In order to tolerate some proxies non-availability, (k, n) threshold proxy signature schemes were proposed in such a way that any subset of k proxy signers in a group of n proxies can collaborate to build a valid proxy signing key.

The need for signature delegation schemes arises for example in MANETs. Permanent communications between clients and servers are unfeasible because of the mutable network topology. In order to nevertheless guarantee service availability to all clients dispersed in the whole network, proxy signature schemes have been proposed to use a fully distributed signature service [77]. An original server delegates its signing capabilities to a group of remote members in the network that cooperatively sign messages on its behalf.

In large-scale wireless networks, deploying a centralized Certificate Authority (CA) to manage key certificates is a very hard task because of scalability and communication delay problems. Many works have adopted the use of proxy signature schemes in order to distribute the CA functionalities to a set of nodes in a collaborative manner. Each designated CA server generates a partial certificate and then collaborates with other CA servers to derive valid certificates to requesters by combining a sufficient number of these partial certificates.

Nodes in WSNs are deployed in unattended and hostile environments to sense and report sensitive data concerning critical applications, such as military surveillance and health monitoring. Providing reliable sensed data despite wireless links vulnerability and nodes' resources constraints is challenging.

The use of collaborative signature schemes has been proposed to prevent the impact of false data reported from malicious sensor nodes. In [78], authors use a threshold elliptic curve cryptography signature scheme that monitors false data emanating from compromised sensor nodes. A reported message should be signed by k distinct sensor nodes before reaching the core node in order to be

considered as valid. Traveling along the full path, reported messages are cooperatively verified by intermediate nodes and signed again in case of agreement. The global verification phase is performed at the core node which verifies the validity of the combined received signatures of all participating nodes. In [79], an efficient collaborative signcryption scheme is proposed to monitor alert messages reported by sensor nodes deployed in a certain area. Each node has a share of a local private key and produces a partial signature during an alert message process. Then a designated sensor node takes in charge the combination of all these valid partial signatures from different participating nodes. If a sufficient number of nodes have cooperatively executed signcryption, it generates a final signcrypted value and transmits it to the base station.

3.2.3. Collaborative radio services

The unpredicted partitioning of wireless networks caused by loss of nodes connectivity or sparse node density leads to unreachable groups of nodes and affects the overall network connectivity. Two nodes belonging to separated groups are not able to communicate with each other since the route between them is interrupted and traditional multi-hop communications cannot restore connectivity. Therefore, a solution is to increase the radio transmit power of a delivered message to reach a disjointed group of nodes. Collaborative transmission has been suggested as a solution to overcome broken links and connectivity problems in multi-hop wireless networks.

The concept of collaboration in radio transmission field has been first introduced by Sendonaris et al. in 1998 [80] for cellular mobile users. In each cell, a user is responsible for transmitting not only its own signal, but also the data of its neighbouring users, which it can detect. The cooperation of in-cell users increases the uplink capacity to achieve a higher data rate.

This concept has been extended to be considered for cooperative transmission in wireless networks. The principle is similar and consists in combining the transmission power of a group of nodes in order to attain a higher transmission power and attain otherwise unreachable zones. Nodes collaborate by transmitting identical symbols at the same time to stack up the transmitted waves on the physical medium. With the sum of waves, the source can reach far destinations. Different cooperative transmission approaches have been proposed in the literature [81], [82]. With wave cooperative transmission scheme [83], nodes receiving a message at the same time repeat it together once to increase the power transmission range. The concept was later extended to tackle the problem when there is only a single node in the initiator's radio range to receive the emitted message. In this case, repeating only once the message may not suffice to achieve the desired power transmission. For this reason, an accumulating transmission scheme has been introduced in [84]. This new alternative proposes that nodes, upon the reception of a message, repeat it cooperatively several times. Hence, even a single node can collaborate with the initiator to get a higher power transmission with the summation of energy and reach an otherwise unreachable node. This technique is heavy in terms of energy consumption since assisting nodes have to retransmit the same message several times. Other cooperative transmission schemes that alternate between multi-hop and accumulative cooperative transmission phases have been proposed such as in [85] and [86]. This hybrid design aims to use multi-hop communication wherever possible and thus to reduce the energy cost of accumulative transmission phase. This scheme offers the highest connectivity level but seems to be complex for implementation in networks with unexpected node behaviour. Assisting nodes have to be aware of the network topology in order to be autonomously able to alternate between multi-hop and cooperative communication phases. This may be only suitable for sparse networks settings and for predictable scenarios.

Cooperation has been also exploited to overcome signal fading problems resulting from multipath propagation in wireless networks [87]. Collaboration is achieved through spatial diversity by allowing multiple users to collaborate and relay each other's messages, developing multiple transmission paths to the destination. Cooperative transmission has been also investigated in resource constrained wireless networks to enable nodes with a single antenna to exploit spatial diversity in order to improve signal quality [88]. Contrary to what happens in conventional multiuser systems, cooperating nodes make their channel resources available to enhance the transmission quality of each other's messages. Each user can act as the source node in a typical collaborative scheme while other users serve as relay nodes. Various collaborative protocols have been proposed based on this concept to advance

communication quality in wireless communications. Examples are Amplify and Forward (AF) [89], Decode and Forward (DF) [87], Compress and Forward (CF) [90], and Coded Cooperation (CC) [91]. AF and DF are the most common cooperative schemes due to their simplicity. With Amplify and forward (AF) scheme, a group of relay nodes receive a signal from a source and simply retransmit it to the destination without decoding it. It is also referred to as a transparent cooperation. With decode and forward scheme (DF), relay nodes are more involved. They decode the received message, re-encode it to enhance error protection and retransmit it as a new message. Upon reception of multiple signals from the source and the cooperating nodes, the destination combines them and recovers the original message. The advantages of these cooperative schemes often depend on the availability of reliable inter-user links. The benefit of AF scheme relies on the quality of the relayed signal since cooperating nodes amplify both the signal and the noise received from the source. Likewise, in DF scheme, an assisting node can decode and relay the message only if it is able to receive reliably the original message from the source.

The outage probability of a transmission within a cooperation process caused by the quality of inter-user channels has motivated researchers to propose partner selection protocols [92]-[95]. These protocols, also referred to as selection cooperation schemes, aim to assign a set of relay nodes to source nodes among a group of potential nodes, and this depending on a figure of merit that takes into account channels conditions and available resources at the relays. In a first coordination phase, all potential relays receive an emitted signal from the source and process it to the destination. At this stage, relays with poor channels are detected and retracted from the pool of assisting nodes while good participants are retained as adequate to cooperate and assist the source transmissions. In a second phase, only selected relays participate to forward the source's messages to the destination.

In order to offer efficient resource utilisation for these cooperative schemes, flexible power allocation techniques have been applied among cooperating nodes [88], [96]. The source and relays coordinate by exchanging mutual information in terms of actual transmission power and channel state. Then, each node adjusts its power allocation so as to minimize the total power allocation required to achieve the desired transmission rate. Opportunistic cooperative transmission schemes [97], [98] propose to dynamically select among all available protocols the cooperative protocol that achieves the minimum total transmission power.

3.3. SYNTHESIS

We assume in this section that the different network services presented in the previous section can be used concurrently. An example of a packet delivery involving collaboration in the fields of radio transmission, routing and security is provided below in Fig. 22.

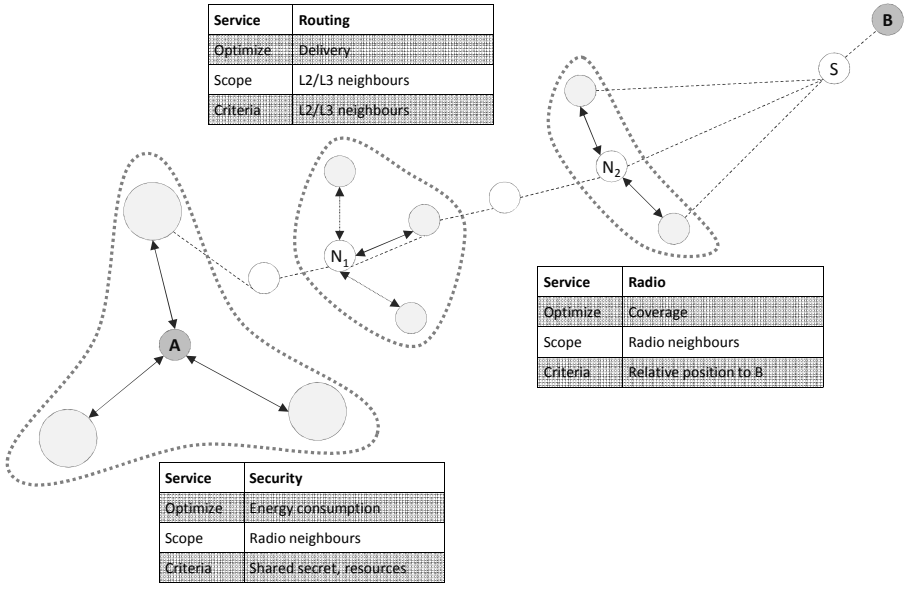


Fig. 22. Involvement of multiple collaborative networking services in a single packet delivery.

Figure 22 is an example of successive use of collaborative services for a packet sent from a node A to a node B. Security enforcement of the A→B keying algorithm is first performed collaboratively at node A with the assistance of resource-unconstrained nodes. Subsequent routing at each intermediary node, such as N_1, is also a collaborative process that involves the choice between multiple candidate next hops and the delivery through the best one. Finally, collaborative radio synchronisation between node N_2 and a set of its neighbours allows the packet to be delivered to the (distant) sink node S, before eventually reaching B.

3.4. SECURITY MECHANISMS AGAINST INTERNAL ATTACKS

3.4.1. Classification of security mechanisms

As mentioned above, collaboration-based services are vulnerable to several attacks caused by the selfish or intentional untrustworthy behaviour of some assisting nodes. Indeed, nodes, especially those with low capabilities in terms of energy and computing power, may be reluctant to make their resources available to other nodes as part of a cooperative act. Therefore, such nodes may prefer to behave at times in a selfish manner in order to maximize their energy savings. With this behaviour, the selfish node unintentionally prevents the system from working properly. Meanwhile, a node intentionally manifesting a malicious behaviour during a cooperative service is not interested in energy savings but in carrying out an attack with the objective to disturb or even damage the system.

That is why it is important to develop dedicated security mechanisms in order to secure collaborative services, especially when conventional cryptographic mechanisms lack to provide required protection against untrusted cooperating nodes. We consider in this section existing security approaches designed to thwart different attacks against collaborative services.

In the literature, security mechanisms are normally classified as prevention, detection and recovery mechanisms. A *prevention mechanism* is implemented to enhance the security during the execution of a system and prevent an attack from occurring. A *detection mechanism* is used to detect both successful attacks and also attempts to violate the security of the system. This security mechanism is usually followed by a reaction phase, used to exclude the attacker or take further measures to prevent or mitigate a future attempt. A *recovery mechanism* is defined as a technique that ensures the system restoration after an attack has been detected.

With respect to the scope of the collaborative approach we are studying, this classification of security mechanisms can be regarded as a distinction between security-by-design mechanisms and behaviour-analysis mechanisms.

Security-by-design mechanisms refer to techniques implemented during the design of the solution to prevent and/or overcome potential attacks. These mechanisms encompass prevention techniques such as access control mechanisms, which actually deny an attacker to be in position of launching an attack, and also include other implemented techniques such as threshold security, which mitigate the attack in order to guarantee a normal operation of the system even in presence of attackers. These mechanisms run inside the service program and can sometimes slow down the system; however, they cannot be dissociated from the service.

Behaviour-analysis mechanisms refer to detection and response techniques. These mechanisms track the system behaviour and interactions between nodes to detect attack attempts and/or occurrences. Once a security anomaly is detected, a reaction mechanism is launched to take security and service repair measures. Security measures include attacker exclusion and punishment techniques. Service repair measures involve recovery mechanisms, such as restoring the firmware of a compromised node to factory default settings. They also consider cognitive techniques used to re-adapt the service to deal with new threats and environment conditions. These mechanisms run along with the service process and can be handled either by the node itself or by another centralized entity. In the literature, behaviour-analysis mechanisms are commonly designated as **trust-based mechanisms**. This terminology will be used for the rest of this document.

Both of security-by-design and trust-based mechanisms complement each other and are designed to be applied together in order to establish a safe environment of cooperative entities. Security-by-design mechanisms (threshold secret key distribution using the Lagrange interpolation and error correction

scheme respectively for the key agreement and the key transport modes) are already taken into account in the design of our collaborative solution, enabling the recovery of the session key even in case of node misbehaviours. Trust-based mechanisms have to be considered as a next step to detect untrusted elements. At this stage, we review existing security solutions proposed to deal with internal attacks in collaborative services in light of the above classification.

3.4.2. Security-by-design mechanisms

3.4.2.1. Collaborative routing services

Several security implementations have been incorporated in routing protocols to stimulate cooperative behaviour among malicious and/or selfish nodes and thwart attacks in wireless networks.

Most of these implementations are based on payment models such as in [99]. These systems provide economic incentives for cooperation. Cooperating nodes are rewarded with virtual currency. The source node has to pay to transmit a packet and assisting nodes are rewarded upon packet forwarding. It is thus advantageous for nodes to cooperate in the packet forwarding process in order to be able to send their own packets. The payment-based model has been firstly introduced by Buttyan and Hubaux [100] in the form of nuglets and establishes a virtual trade market between nodes to enforce cooperation. Nodes are stimulated to cooperate in packet forwarding because they need to earn nuglets. To ensure that nodes do not forge payments, [101] proposes the use of tamper-proof hardware to secure the credit exchange. Nuglets are transported within the packet and stored in a specific encrypted header called secure module. The tamper proof based mechanism encourages nodes to cooperate; however, it requires deploying a new secure module using cryptographic operations in each node, which introduces a significant implementation complexity and communication overhead. A good economic incentive mechanism has to provide lightweight overhead in the network and secure credit exchange. During the last decade, several solutions have been proposed aiming to secure the payment process and to efficiently implement it in the routing protocols. In [99] a central bank based mechanism named Sprite proposes to replace the use of a specific hardware module with a central controller called Credit Clearance Service (CCS). CCS stores and manages nodes accounts in the system. Nodes periodically report receipts resulting from their cooperative actions to the central controller in order to update their accounts. Sprite model motivates node cooperation without need for hardware implementation; however, it induces an extra communication overhead in the network due to the large number of reported receipts. In [100], proposed solutions still focus on securing the payment packet routing process and rely on centralized authorities and public key infrastructure to manage the credit exchange between nodes. Although these solutions claim achieving security requirements, the heavy computational cost and extra overhead they introduce degrade the network performance and drain nodes resources. For this reason, further solutions have focused on reducing the heavy cost induced by payment-based models especially for resource-constrained networks. In [104] and [105], a payment aggregation mechanism is introduced to generate a receipt for multiple packets instead of delivering a receipt per packet. In [106] and [107], a probabilistic payment technique is proposed to reduce the overhead resulting from the large number of reported receipts in the network. Other solutions addressed the question of how much a cooperating node should be paid for forwarding packets. In Ad hoc-VCG [108], cooperating nodes are rewarded according to the energy they consume to relay the packets. However, a node may cheat on its real cost in order to maximize its payment. Based on the concept of game theory, the Vickrey-Clarke-Groves (VCG) model was designed to ensure that the profit of each cooperating node is maximized when it reveals its true cost regardless the declarations of other nodes. Thus, it is no longer profitable for nodes to cheat on their cost during a routing protocol. In [109], the pricing mechanism increases the rewards for cooperating nodes proportionally to the load of the network. In [110], nodes rewards are assigned based on their available bandwidth and power level. In [111]-[113], authors address the case where a group of colluding nodes work together to maximize their benefits and propose collusion-resistant payment mechanisms based on standard concepts for collusion resistance in game theory.

Apart from payment based models, other mechanisms have been designed for routing protocols to maintain a reliable packet forwarding process in presence of misbehaving nodes. These solutions take advantage of route diversity (multiple routes between nodes) in wireless networks to apply diversity coding [114]. Exploiting route diversity has been firstly introduced in [115]. Authors highlight the

need for specific mechanisms to ensure routing service availability in case nodes are compromised, especially since traditional cryptographic mechanisms are not effective by themselves in these situations. The proposed mechanism consists in transmitting redundant data through additional routes in the network for error correction. Hence, even if the primary route is compromised, the receiver can recover the original message using redundant data received from additional routes. Transmitting through multiple routes is a robust prevention mechanism to cope with malicious behaviours. For this reason, multipath routing approaches become an attractive research field in wireless networks to enhance transmission reliability and provide fault-tolerance against attacks and node failures [116]-[117]. The solutions detailed in [118]-[120] rely on specific metrics such as energy consumption and node stability to select routes between a given source and a destination in order to increase the reliability of packet transmission.

3.4.2.2. Collaborative security services

Traditionally, proposed collaborative security approaches either implicitly or explicitly assume that cooperating nodes are trustworthy. Basic schemes such as polynomial based Shamir scheme and proxy signature scheme of Mambo suppose that all players are honest and an adversary is assumed to be unable to disturb the system. Afterwards, security analysis for these collaborative approaches started being considered in the literature, since trusting all players is impossible in practice.

In order to prevent dishonest behaviour of some participants, a number of robust secret sharing schemes have been proposed. Verifiable secret sharing (VSS) schemes [121]-[123] were designed to deal with malicious players. These schemes guarantee a correct reconstruction of the secret in case a dishonest dealer or participant provides fake shares respectively during the dealer distribution or the combiner reconstruction phase. Proactive secret sharing solutions have been proposed to overcome mobile adversary attacks [124]. In fact, a mobile adversary may take profit from the long lifetime of the secret so that it attacks a sufficient number of servers one by one to learn and destroy the secret. We consider that our collaborative key establishment schemes are less vulnerable to mobile adversary than secret sharing schemes since assisting nodes store secret shares for a short period of time and are to delete them after the key exchange. The basic defence against this attack is to renew the secret periodically; however this could be impractical for long-lived information such as cryptographic master keys or sensitive data files. Proactive schemes [125]-[126] protect the secret sharing against these attacks by periodically refreshing shares while keeping the same secret. Since attacker capabilities are likely to increase over time, it will become simpler for it to compromise many participants in a short time. To counter this threat, authors in [127] have introduced the changeable threshold secret sharing scheme, proposing to adjust the threshold parameters according to the environment reliability. Thereafter, several secret sharing schemes have investigated the flexible change of the threshold value in their solutions. First proposals [128] and [129] required establishing secure channels between the dealer and participants to redistribute shares corresponding to a new threshold value. Then, more flexible schemes have been proposed, eliminating the dealer presence during the threshold update process [130]-[131].

Likewise, other authors have reviewed the security of proxy signatures schemes and proved that they are insecure against various insider attacks [132], [133]. This was not surprising, since basic constructions of Mambo's proxy signature scheme overlooked insider misbehaviours emanating from the original signer and proxy signers.

The concept of threshold proxy signature has first been proposed by Kim et al. [76], based on the secret sharing schemes. K. Zhang [134] proposed a new (t, n) threshold proxy signature scheme. The common idea is that a proxy signature key is distributed among a group of n proxy signers in a way that at least t proxy signers can cooperate to sign messages on behalf of the original signer.

After that, Sun in [135] revised the security of Kim and Zhang's scheme and proved that it suffers from some weaknesses: indeed, a proxy signer can repudiate a signature creation since the proxy signature does not provide any authentication information about the identity of the signer. In order to solve the problem of unknown signers, Sun improved Kim's scheme and proposed a non-repudiable proxy signature scheme with known signers so that a verifier could identify the actual signers and determine whether the group signature key was generated from a legitimate group of proxy signers. But, Hsu et al. in [136] revealed that Sun's scheme suffers from a collusion attack that any group of t

proxies or more can modify the threshold strategy to a new (t', n) one. They proposed a new non-repudiable threshold proxy signature scheme with known signers.

Shum et al. [137] introduced a strong proxy signature scheme with a proxy signer privacy protection. A proxy signer can sign messages on behalf the original signer while protecting his privacy against outsiders. Lee in [138] showed that Shum's scheme lacks the property of strong unforgeability, since either the original signer or another third party can play the role of proxy signers and generate a valid proxy signature. Zhang et al. [139] proposed a new proxy ring signature to resolve this problem. So far, many proxy signature schemes [140]-[142] have been proposed pointing out security weaknesses of some previous schemes and proposing countermeasures to improve them.

3.4.2.3. Collaborative radio services

The vast majority of research studies on cooperative transmission focused on improving the efficiency of signal transmission and reliability in the network, assuming that cooperating relay nodes are trustworthy. More recently, a limited number of studies have considered security issues in cooperative transmission. In [143]-[144] authors highlight the vulnerability of current cooperative transmission schemes to misbehaving relays without proposing special security countermeasures. In [143], simulations are carried out to evaluate the performance degradation of cooperative transmission systems under relay misbehaviours. In [144], the authors assess to which extent a cooperative transmission can outperform a single transmission in the presence of misbehaving nodes. Motivated by the lack of security mechanisms to ensure the commitment of a relay node to the cooperation strategy, recent works provide cooperation incentives in cooperative communications. These studies are inspired by the pricing-based mechanisms proposed for cooperative routing services discussed above, and adapted to the cooperative transmission context with multiple relay nodes. In the same way, cooperating nodes are rewarded with virtual currency. Source nodes make payment to participating nodes for using their resources to relay their packets. Unlike the payment in packet forwarding schemes where prices are fixed and the utility of a relay depends only on its own strategy, payment in cooperative transmission is shared among a set of players participating in the same relaying process. Hence, the utility of a node will depend on the strategies of other relays creating a competitive scenario. In case of one source node and multiple relay nodes, authors in [145]-[146] formulated the interaction between players as a buyers' market, and modeled it as a Stackelberg game with the source node as the leader and the relay nodes as the followers. In game theory terms, Stackelberg model is a strategic game in economics where the leader takes action first and the followers take actions afterwards. The leader knows in advance that the followers perceive its action and takes action considering that fact. Zhang et al. in [147] studied the case of one relay node and multiple source nodes; the market is expressed as a sellers' market. In this mechanism, only the source nodes are players and compete to obtain from the relay node the bandwidth they require. Reference [148] proposed an auction scheme where relay nodes propose prices on their relaying services and allow source nodes to bid on them. Resources allocation for each source node depend on source nodes bids. In [149], authors showed that the above pricing schemes proposed for cooperative transmission only deal with selfish behaviour of players and are vulnerable to cheating behaviour. In other words, a source node can submit a bid higher than its true valuation in order to maximize its profit in terms of resource allocation. Motivated by this weakness, they designed a trustworthy auction scheme based on VCG model which enforces players to reveal their true valuations to maximize their individual profit, thus eliminating the impact of cheating behaviour on the cooperative system performance.

3.4.3. Trust-based mechanisms

All of the above security-by-design schemes aim either to stimulate cooperation between nodes in order to prevent selfish and/or malicious attacks, or to guarantee a proper operation of the cooperative service in presence of attackers. But generally these mechanisms are not able to detect misbehaviours nor to handle ongoing attacks, since they are not designed to trace nodes interactions during the service execution.

Trust management mechanisms aim to track nodes past experiences to detect malicious attacks and selfish attitudes. These mechanisms should also apply punitive measures as a reaction phase after

detecting an attack in order to deter the misbehaving node. Trust-based mechanisms help improve node selection decisions by designating only reliable participants, based on their historical activities.

We have extensively studied existing trust-based models proposed for cooperative services in wireless communications. We have identified that the most proposed models addressed packet forwarding services. Limited related work has been recently conducted in the field of cooperative transmission. No existing work on trust is proposed for cooperative security services.

We describe in this subsection research work on trust-based mechanisms proposed for both routing and cooperative transmission services. We discuss then their applicability for our collaborative key establishment schemes in the context of the IoT.

3.4.3.1. Trust management systems for collaborative routing services

Several trust management systems have been designed for packet forwarding services in wireless networks. The goal of these systems is to detect misbehaving nodes causing routing disruptions, to penalize malicious nodes and therefore to enhance decisions making in the future. Marti et al [150] have proposed the first work that introduces trust and reputation based mechanisms. They recognize that misbehaving nodes in packet forwarding can significantly affect the network throughput and underline the case where malicious nodes accept to relay packets but later do not accomplish the assigned task. To defend against this threat, they make use of the watchdog technique, which consists in monitoring the neighbouring traffic in order to detect misbehaving nodes. They also use the pathrater technique to avoid misbehaving nodes when selecting the most likely reliable route for packet routing. However, this model has not specified any punishment measures against misbehaving nodes.

In [151], a distributed trust model called CONFIDANT is proposed. It considers Dynamic Source Routing protocol and aims at detecting and isolating misbehaving nodes during the packet forwarding. The proposed model takes into account both first-hand and second-hand information to update trust values. In the first-hand information based models, the system relies only on its direct observations and own experiences to update nodes trust values as in [150]. This reflection can be useful when a node is active but when this latter has sparse interactions or its requirements change frequently, it may lack sufficient information to make trust decisions about other nodes. To make both the trust model more robust and the computed trust values more reliable, CONFIDANT extends the previous work to disseminate trust throughout the network. Thereby, it also takes into account indirect experiences and observations reported by neighbouring nodes to evaluate the trustworthiness of relay nodes. Only negative observations are exchanged between nodes, assuming that misbehaving nodes sending false reports will be the exception and not the norm. Obviously the system, with such assumption, is vulnerable to false reports causing the trustworthiness of benign nodes to decrease (***bad mouthing* attacks**). Low reputation nodes are completely rejected from the packet forwarding process. Authors in RRS [152] improved CONFIDANT and introduced a Bayesian model with Beta distribution to explain how actual trust values are computed. Both positive and negative reputation values provided by second-hand information are used to compute a trust value about a specific node. The confidence put in collected reports is integrated as long as the reporting node is classified as trustworthy. The trust metric is used to determine whether the node can be trusted or not to perform an assigned task. The proposed model assigns a higher weight to recent behaviours, considering that a misbehaving node can initially build a high reputation with good behaviours and then remain trusted while misbehaving. SORI scheme [151] proposes another distributed trust-based model to enforce node cooperation in MANETs. Each node in SORI listens in promiscuous mode to packet transmissions within its one-hop range. Trustworthiness is evaluated through ratio between the number of packets a relay node has forwarded and the total number of packets it is assumed to relay. Neighbouring nodes exchange these local evaluations periodically. If the trust value of a node falls under a threshold, this latter is detected and signaled as a suspicious node. The SORI model is more tolerant than CONFIDANT in terms of punishment decisions. A misbehaving node is never completely excluded from the routing path and can continue to increase its reputation value by behaving cooperatively with other nodes.

In [153], the CORE model is proposed. It is a generic trust-based mechanism aiming to detect selfish behaviours for different cooperative services. The watchdog mechanism is implemented to monitor interactions between nodes performing a cooperating service, which is not limited to packet forwarding. The model assigns a global trust value to a cooperating node for all provided services.

Unlike CONFIDANT, CORE mitigates bad-mouthing attacks caused by malicious nodes reporting false evidences to decrease the reputation value of a node. Indeed, it allows only positive witnesses to be propagated in the network, assuming that a node has no advantage to give a false praise about unknown nodes. Nevertheless, this model overlooks the case where nodes collude together by disseminating false evidences to increase their reputation values (called ***ballot stuffing*** **attacks**). CORE does not apply the same measures to punish misbehaving nodes and considers that a selfish node restrained by its low resources should not be penalized like a malicious node deliberately affecting the service performance.

In [154], authors highlight that previous trust models suffer from low scalability since reputation information has to be propagated among all nodes in the network and can be biased when poisoned with false reports. They propose a novel trust-based approach to enforce collaboration in routing services considering these problems. Reputation values are kept local and the node monitors only its one-hop neighbour nodes through direct observations. Once a non-cooperative behaviour is detected, benign neighbours will redirect received packets through another route to avoid the misbehaving next hop node. This latter is implicitly rejected from the network since in turn all of its neighbours will reject its packets as response to its future routing service requests. A cognitive reputation based scheme is proposed in [155] to reinforce routing in heterogeneous wireless communications. To monitor nodes behaviours during path selection, a routing algorithm is created to compute reputation of next-hop nodes based on feedbacks reported by the two-hop neighbours. A feedback is transmitted along other routes different from the forwarding route that contains a hash value of the received data encrypted with the source's public key. In a nutshell, the source learns to classify the behaviour of the one-hop neighbours from the testimonies of its two-hop neighbours. Authors recognize that feedback information can also be vulnerable to unreliable paths and propose to send redundant feedback information through multiple disjoined paths. Reputation values are computed locally at each node using the Beta Bayesian approach.

RFSN [156] is the first trust-based model proposed for wireless sensor networks to monitor sensor nodes interactions. Each node maintains trust values of other nodes using its direct observations from the watchdog mechanism and second-hand information from other nodes observations. The computation of trust is based on the beta distribution giving more weight to latest observations. Like CORE, the proposed system allows only positive observations to be propagated, making the bad mouthing attack impossible. It relies on the trustworthiness score of the witness node to weigh its reports in order to overcome ballot stuffing attacks.

An agent-based trust model for wireless sensor networks is presented in [157]. It allows to move the heavy computational and storage cost required to handle trust at constrained sensor devices to dedicated agents in charge of cooperation management inside the network. The proposed system claims to be safe from bad mouthing or ballot-stuffing attacks, assuming that the deployed agents are trusted-third parties and would not engender these types of attacks.

3.4.3.2. Trust management systems for collaborative radio services

Authors in [158] were the first to design a trust-based model for cooperative transmission in 2007. In the proposed scheme, each node maintains link quality information between itself and its neighbour nodes. The link quality is computed using the beta function model and stored as a trust value. A node checks the cyclic-redundancy-check (CRC) and the signal-to-noise ratio (SNR) of the received signal and can infer a trust value of this link. This trust value is estimated directly from its own observation and is therefore considered as direct Link Quality Information (LQI). On the other hand, each node also receives indirect LQI reports, estimated by other nodes. Gathering this information, a trust manager module implemented at each node detects links with low quality and disregards them during the relay selection. The proposed approach allows malicious attacks detection at the destination by putting in opposition the observed link quality from real data transmission and the estimated reports from other nodes. Lying relays are penalized by reducing their associated weights to zero during the signal combination.

Authors in [159] show that the above work bases only on the number of successfully received packets in its trust computation process and does not consider the channel condition and relay selection policies. They make clear that unsuccessful packet transmissions from a relay node are not always the result of a malicious behaviour, but can be also due to other factors such as channel

congestion or packet overflow. They introduce a new distributed trust approach that modifies the trust establishment method and takes into account the channel state information and relay selection decision for signal combination at the destination.

Some studies have been proposed to provide trust management mechanisms in cooperative communication without explicitly designating them as trust-based models. In [160], authors revealed the vulnerability of cooperative wireless communications to garbled signals generation by compromised nodes. A cross-layer framework for tracing malicious relays is proposed. The basic idea from the tracing scheme is that the source inserts pseudorandom tracing symbols along with the initial signal before transmitting it to the relays. These tracing symbols can be extracted and verified by the destination only to detect the ground truth of each relay node. Indeed, the destination is the only node sharing the tracing key with the source.

In [161], a smart destination which analyses relay signal prior to applying signal combination is considered. The analysis phase consists in computing correlation between signals received from the source and relays. Since the malicious behaviour significantly decreases this correlation, the destination becomes able to detect the responsible relay node. Authors expect that this detection mechanism can be further explored as a part of a global trust management framework, which also implements reaction mechanisms, imposing penalties to misbehaving partners.

In [162] authors highlight that existing work on misbehaving relays detection requires perfect channel state information that may not be always available. Based on this weakness, they redesign the malicious relays tracing technique described in [160] in the absence of instantaneous channel information at the destination. This lack of information makes the destination unable to demodulate the received tracing symbols in order to detect malicious relays. In this work, authors propose to identify misbehaving relays by sensing the distribution of the phase rotations of the tracing symbols. As long as relays behave in a cooperative way, these tracing symbols undertake similar phase rotations. Simulations show that the proposed scheme has considerable detection performance compared with existing work requiring perfect channel state information knowledge.

3.5. Design Decisions for the Application of a Trust Management System in the Context of our Collaborative Keying Solutions

As network aspects have changed with the advent of the Internet of Things, new design decisions are to be taken into account for the design of a trust management system, in order to fit additional requirements of the IoT and make viable decision makings for our collaborative key establishment protocols.

By essence, the role of a trust model is to assist network entities in the decision making process. As such, it is reasonable to expect that a trust model will perform better if it processes more input data, being able to issue a recommendation out of a set of diverse gathered elements that all help to build a clearer model and assessment of the situation. Yet, this recommendation must also be adapted to the context of the node requiring assistance for decision making. The rule of thumb in trust model design can therefore be seen as an instantiation of the famous "think globally, act locally" paradigm. Among the various elements a trust model is made up of, some have to be considered together, some others have to be kept separated and yet some others have to be put in relationship with each other through thoroughly designed weighting functions.

Considering reputation and trust separately is maybe **the first design decision** that should be taken into account. While reputation refers to the good or bad behaviour of an entity, trust refers to the ability or inability of an entity to fulfil a certain function. Most of prior trust models do not make the difference between reputation and trust metrics. In this thesis, we recognize that a node under the context of the IoT may change from a context to another due to its variable resources capabilities (energy exhaustion, energy harvesting) and variable status (mobility, processor availability). So the fact that a node has a good reputation when it is in a specific context gives no information about how much it can be trusted to provide assistance for a cooperative service when changing to another context. A node classified as honest could behave well with 80% of available resources. Yet, there would not be any guarantee that the same level of benevolence would be obtained for the same service in another situation where only 20% of its resources would be available. To that aim, additional

parameters are to be considered such as energy resources and availability of the scored entities, evaluated by the trust model. If one takes the example of a node candidate for assisting in a security service, it becomes perfectly clear that the computing power of this node has to be evaluated, along with its ability to remain present and available during the service exchange.

A second important design decision is that a trust model should be able to monitor behaviours according to different functions, considering for which service the assistance of a given node was required. Demanding aspects may change from one service to another. A node trusted to provide assistance for a simple service (e.g. routing a packet) may not necessarily behave well for a resource-demanding service (e.g. signing messages). The CORE model [160] proposes such a functional trust management system. However, it eventually assigns a single, global, trust value to a node. This simplification comes at the expense of a lack of flexibility and adaptability to complex malicious patterns. Indeed, when considering a global trust value for all services, a subtly behaving malicious node may show a high level of benevolence for a non-demanding service while behaving poorly for a resource intensive service, which would allow it to keep an overall fair trust value.

Though the contextual and functional environments of each observation should be kept along with the observation report itself in order to satisfy the second design decision, there is of course a need to combine distinct reports (positive/ negative evaluations issued from different nodes without any initial restrictions), to make the information complete when a candidate assisting node has to be evaluated. **A third design decision** would therefore consist in defining a rigorous method to perform this combination. An evaluation being obtained from a synthesis made over a plurality of individual reports, defining the combination operation amounts to defining a weighting algorithm that gives an optimal weight to each individual report. This weight reflects the relative confidence put in each report, with respect to its representativeness of the situation in which the evaluation is performed. As such, the weight of a report changes depending on the situations where this report is used (time, context, type of service). The report weight also reflects the confidence that is put in the report originator, which leads us to **a fourth design decision**.

The confidence in a report originator depends on the evaluation of the originator recommendation quality. Currently, there is no trust management system in wireless communications that handles reports received from witness nodes basing on their quality of recommendation. Prior trust models propose either to only consider direct experiences while overlooking reports of other nodes to avoid false witnesses or to base on the trustworthiness score of a node when assisting a service to estimate its trustworthiness when providing reports. The first case would be efficient for a node involved in numerous transactions with other peers; however, a node having only sparse interactions with assistant peers or whose requirements are changing frequently may lack information to make trust decisions about other nodes. In the second case, mixing two trustworthiness scores together would encourage a node to take advantage of this fact and send correct recommendations while misbehaving as a service assistant. It would then remain overall trusted, since its bad behaviour in service setup would be compensated with good behaviour in recommendations. The score given to a node to evaluate its recommendation quality should thus be kept independent of the score evaluating it as an assistant in a collaborative service.

The four design decisions listed above (trust / reputation distinction, combined {function, context, observation} storage, weighting factors rigorous definition and update, service assistance and recommendation quality separation) will be carefully considered for the design of a generic functional trust model. These design decisions can be applied in general and do not depend on a specific topology. In addition to these generic principles, a few other incidental ones have to be taken into account. Deciding when to trigger the operation of the trust management system and where to instantiate this system are choices that are much more dependent on the studied topologies and the capabilities of the nodes they are made of. In what follows, we answer these 'when' and 'where' questions with respect to a heterogeneous wireless topology that includes highly resource-constrained nodes.

The trust information can be computed on demand (whenever a node has to rely on collaborative peers) and delivered to the requesting node at that moment, or it can be computed on a regular basis and be propagated throughout the topology. In the heterogeneous topology considered in this thesis, the former option appears much more viable for two reasons. First, a real-time trust information flow would result in communication overhead, detrimental to network performance as well as to

constrained nodes battery life. Second, unsolicited trust information would have to be stored for subsequent use, which memory-constrained nodes may not be able to afford. The storage cost of trust information chunks would be all the more complex when these chunks are multidimensional (our second design decision forbids globalizing a node's trust value) and likely accompanied with cryptographic authentication MACs.

The choice between centralized and decentralized instantiation of the trust management system must take into account its complexity in terms of trust computation formulas and processed information quantity. Here, our second and third design decisions lead us to favouring a centralized approach, wherein a central server would handle the complex node evaluations, based on a wide range of reports.

Offloading the charge from the most constrained nodes by taking profit of a much more powerful entity is not the only advantage of the centralized approach. Having to send its observations to the central server instead of sending them to other nodes, a malicious node would not be in position to send false reports to specific victims only, in order to fake their decisions. With a central entity responsible for trust management, it becomes a common profit for all nodes to provide reliable evidences since false ones can globally affect decision making at the central entity, and could eventually be detrimental to the attacker itself. Finally, relying on a central entity reduces the information asymmetry by letting a node with a global view of the network compute the trust value of all nodes.

Though centralisation appears as the right architecture scheme with respect to trust management systems for resource-constrained entities, local parameters depending on initial network setup must not be neglected. Relative and absolute locations of a candidate assisting nodes can be provided as examples of these configuration parameters. The relative location of the candidate assisting node to the requesting node may indicate whether the former shares a pairwise key with the latter or belongs to the same multicast group (clustering). The absolute geographic location or at least an estimation of the candidate node's location within the considered topology could be required as well and is actually needed in some collaborative signature schemes. Both location information elements help the central trust management system to issue relevant recommendations.

To summarize we provide the table 18 below.

Table 18. Assessment of trust model design decisions.

Prior trust models practices	**IoT requirements/constraints**	**Trust model design decisions for our collaborative keying approach**
Mix trust and reputation metrics together.	Variable contexts of IoT nodes and different resource capabilities.	Evaluate the trust level of a node by taking into account additional parameters concerning its current context.
Define a global trust score for all assisted services.	A node trusted to provide assistance for a lightweight service is not necessarily trusted to assist for a service demanding more resources and increased availability during the service execution.	Design a functional trust model that takes into account the specific demanding aspects of the assisted service when assigning a trust level.
Restrict the reception of certain reports from witness nodes to avoid bad mouthing and ballot stuffing attacks.	Lack of information to take trust decisions due to the sparse interactions of constrained IoT nodes.	Consider all received reports and past interactions in making trust decisions by defining new methods to perform the combination and bypass the underlying attacks.
Do not separate received reports from witness nodes basing on their quality of recommendation.	IoT nodes belong to different groups and may provide false witnesses since they do not work towards the same goal. Trusting a node as an assistant node does not imply trusting it as a reporting node.	Weight reports basing on the trustworthiness of nodes as reporting nodes.
Consider both centralized and decentralized instantiation of the trust management system.	Most of IoT nodes are characterized by low capabilities in terms of both memory and computing resources which make them unable to support the complexity of trust computation and data storage.	Favor the centralized approach to offload the underlying charge from constrained nodes and reduce the communication overhead between nodes.

3.6. CONCLUSION

In this chapter we explored the different manners of addressing collaboration for diverse networking services. In these services, collaboration ensures a much better operation of a cooperative topology than the mere juxtaposition of individual, self-oriented decisions. Machine to Machine (M2M) and Internet of Things (IoT) architectures accentuated the collaboration trend that was initiated in Wireless Sensor Networks (WSNs). They did it not only by involving wider architectures but also by adding heterogeneity, resource capabilities inconstancy and autonomy to once uniform and deterministic systems. Indeed, M2M and IoT nodes have a greater need to collaborate with each other when they are constrained and/or diverse, when they share a common rare resource - such as our collaborative solution for key establishment - or when they are expected to feature an adaptive / cognitive function such as self-healing.

However, we highlighted that the emerging advantages of collaborative approaches could be hindered by their inherent exposure to internal attacks: during a collaborative task, a single or a group of malicious node(s) can disturb the proper operation of the entire system.

The internal attacker may be prevented from performing harmful actions through a careful design of the collaborative protocol that may, for example, require redundant processes to be performed by different nodes or a threshold security procedure enabling a proper operation of the system in presence of this attacker. These prevention and recovery techniques are designed as security-by-design mechanisms. Or it may be detected as malicious through an analysis of its behaviour, which becomes more complex with large-scale heterogeneous architectures such as M2M and IoT. This detection technique is designed as a trust-based mechanism. These two ways of mitigating attacks against collaborative schemes are complementary and should coexist together in order to safely manage collaboration inside a group of nodes. Among these security mechanisms, our solution presented in the previous chapter did provide security-by-design. However, its reliance on a trust management system for selecting trusted elements and assessing cooperating nodes behaviours was only implicitly mentioned at this stage.

By studying existing trust management systems, we could pinpoint gaps in current approaches with respect to the context of our keying solutions. Hence, we identified a set of relevant design practices that oriented the conception of a novel trust model, which we propose in order to build a generic functional trust management system, fitting the requirements of our solution and the IoT environment. This will be the subject of the next chapter.

Chapter 4: TRUST MANAGEMENT SYSTEM DESIGN AND PERFORMANCE ANALYSIS

4.1. INTRODUCTION

In this chapter, we propose a novel Trust Management System (TMS) for the IoT that involves nodes with different resources capabilities. As compared to legacy Internet, the Internet of Things exhibits a greater autonomy, instantiated in the form of multiple *self-** functions. The wide majority of TMSs proposed for wireless networks are today bound to a single function. As such, they cannot use past experiences related to other functions. Even those that support multiple functions hide this heterogeneity by regrouping all past experiences into a single metric, which strongly degrades the quality of results. They consider that a node will behave fairly or maliciously as a whole, but do not take into account the current status of the node (available resources). Neither do they separate reports based on the demanding aspect of the services they refer to.

Based on a set of guidelines identified in the previous chapter for TMS design, we propose a context-aware multi-service trust management system that manages cooperation between nodes for establishing a community of trusted elements assisting each other. This system is able to induce from a node's past behaviours in distinct collaborative networking services, including our collaborative key establishment services, how much trust can be put into that node for accomplishing a required task. Eventually, only the best partners with respect to a sought collaborative service are proposed to a requesting node. Our system quickly identifies poor/misbehaving nodes, even in the presence of wrong or malicious recommendations.

The design description of the proposed TMS is detailed in the section 4.2 of this chapter. Section 4.3 presents then the technical implementation of the proposed solution. We analyse in section 4.4 the simulation results we have obtained, which prove the effectiveness of the proposed trust model and its robustness against attacks. Finally, we conclude this chapter in section 4.5.

4.2. PROPOSED TRUST MANAGEMENT SYSTEM

4.2.1. Overview

The main objective of the proposed solution is to manage cooperation in a heterogeneous wireless topology involving nodes with different resources capabilities, in order to establish a community of trusted elements assisting each other.

The operation starts with the trust manager assigning cooperating nodes, or "proxies", to requesting nodes in order to assist them for the collaborative services they are demanding. After having obtained assistance, each requesting node sends a feedback to the trust manager, specifying its satisfaction level about each participating proxy. By analysing the received reports, the trust manager learns about the results of its last assignment decision. It becomes able to detect misbehaving nodes and to refine its selection in the future.

In the considered architecture, the trust manager is thus the component that is in charge of storing the experiences of nodes in the network and making global trust decisions. The other nodes that exist in the network play either the role of service requesters asking for assistance from other nodes to accomplish a service, or proxies (P_i) designated by the trust manager to assist for specific services.

A description of the different phases of the proposed model is presented in the figure 23 below. This model involves a cyclic succession of operations wherein: 1) the trust management system (trust manager) obtains information about the trustworthiness of the available proxies, 2) the trust management system issues recommendations about proxies to a requesting node that intends to set up a collaborative service, 3) the requesting node relies on the collaborative service provided by the recommended proxies, 4) the requesting node assesses the quality of each individual service provision from each assisting proxy and 5) the trust management system learns from its past operation by performing self-updates intended to improve its future operation. These five phases our proposed model is made up of are reviewed in the next subsection.

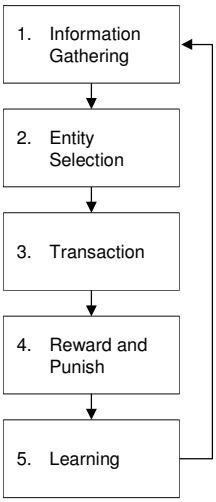


Fig. 23. Proposed model phases.

4.2.2. Operation phases

4.2.2.1. Initialisation and information gathering

At the beginning of the lifetime of the network, the proposed model is initialized with all nodes being assumed to be trustworthy and well-behaving. In a controlled architecture (e.g. wireless sensor network), all nodes are indeed supposed to be verified for failures before deployment. Only once the network becomes operational, nodes may happen to become compromised and their trustworthiness levels will therefore have to be adjusted with respect to their behaviours. Uncontrolled architectures (e.g. Internet of Things) offer fewer assumptions about the initial status of nodes. Yet, here also, initially assuming that all nodes behave trustworthily makes the system converge quicker to the state where it is able to identify the nodes for which this assumption was false. This requires, however, that the number of trustworthy nodes exceeds 50% of the overall nodes population.

Before being able to produce trustworthy results, a trust management system has to gather enough information from the network, during a so-called bootstrapping period whose precise definition depends on the requirements on the recommendation quality. A trust management system is indeed expected to produce better results over time: a compromised node may remain unnoticed for a while (especially since the initialisation process will have set its initial trustworthiness level to the maximum value); but it will be more easily spotted if it gets involved in a large number of transactions, all of which are poorly rated.

The bootstrapping period can be long, since a true assessment of nodes behaviours needs to be carried out over multiple transactions. In order to minimize the bootstrapping period, the trust manager may be involved in the setup of the trust management process by targeting some nodes and inducing dummy artificial interactions between them (in essence probing the nodes), in order to accelerate the rating of their trustworthiness. However, this process could be exploited by intelligent attackers, who would pretend to be benevolent during the bootstrapping phase only: the bootstrapping dummy transactions would therefore have to be made non-distinguishable from the subsequent legitimate ones.

When a service is provided, the requesting node is able to evaluate the behaviour of each assisting node as either positive or negative, depending on whether it has accomplished its assigned task properly or not. It delivers then the evaluation to the trust manager.

The evaluations are stored in the trust manager and used as inputs for the trust management system. In order to make assisting nodes recommendations more accurate and specific there is a need to store, along with the evaluation score, additional contextual metrics concerning the type of executed service, namely the time of execution and the current state of the evaluated node (aging, resource capacity, etc.) It is indeed important to know in which circumstances the cooperating node has obtained the reported evaluations.

Contrary to what is proposed in the literature, our trust model proposes an objective mechanism providing dynamic trust ratings for the same node, adapted to the different behaviours exhibited in different contexts. This mechanism states that an evaluation report is accompanied with a set of contextual parameters, as

follows. A report R_{ij} referring to the j^{th} report sent to evaluate the quality of the service provided by an assisting node, or proxy, P_i is therefore made up of the following information:

- $[S_j]$ (*Service*): the service for which the node P_i provided assistance.
- $[C_j]$ (*Capability*): the capability of node P_i when assisting the service.
- $[N_j]$ (*Note*): the score given by the requester node to P_i for evaluating the offered service. $N_j \in \{-1, 0, 1\}$. The score '1' corresponds to a good-quality service; the scores '-1' and '0' respectively correspond to a bad-quality or not-provided service and to a partially acceptable service.
- $[t_j]$ (*Time*): the time at which the service was obtained.

4.2.2.2. Entity selection

Upon receiving a request from a node asking for assistance, the trust manager starts the entity selection process to return a set of trustworthy assisting nodes to the requester. We propose a step-by-step selection process. This process is the most important of the trust management system. It is made up of five consecutive steps that all happen within the trust manager.

Step 1: Restriction of the set of proxies P_i

The system first restrains the set of nodes by selecting the potential candidates. This selection depends on the requirements of the service. A security service such as our collaborative key establishment scheme requires that the requesting node shares a symmetric key with each assisting node, which typically narrows the set of acceptable proxies. Likewise, the need for lightweight communications may also require that all assisting nodes belong to the same multicast group and can therefore be contacted simultaneously through the (cheap) sending of a single message.

In the case of signature delegation schemes, the requesting server looks for assisting nodes dispersed in specific locations in the network in order to sign messages on its behalf, and hence to ensure service availability to all of its clients. In radio transmissions services, neighbours in the same radio range are the only possible candidates for assistance.

Step 2: Restriction of the set of reports R_{ij} for each proxy P_i

After the prior selection, a set of nodes are designated to compete for the final selection. In order to rate the trust level of each of these candidates, the trust manager needs first to narrow the set of collected reports about each node independently. The most meaningful reports are those that pertain to the same context as of the current request: ideal reports would be pertaining to the same service that is being requested; they would also have been issued when the evaluated nodes were in the same status as of the new request moment.

It is very likely, though, that the system will not find enough such ideal reports to calculate the trustworthiness of a node in a specific context. This may happen either because the candidate node has not yet been evaluated for the current requested service, or because it was in a different condition when evaluated for the same service. To resolve the problem of this lack of information, we proposed to calculate context similarity.

The graph below in figure 24 describes how we restrain the set of potential reports needed to evaluate the trust level of a node by considering the principle of context similarity in terms of type of service (x-axis) and node capabilities (y-axis). This two-dimensional context representation assumes that one is able to quantify the two values it relies on. Node capabilities can easily be quantified, for example as a percentage of node resources in terms of processing power, memory and/or battery level. It is more complex to quantify the former term, namely context similarity in terms of type of service, since multiple collaborative services exist that share little in common.

We consider that an adequate metric for assessing service similarity is the amount of resources that are required to run a given service. Within the resources that can be measured, we recommend to consider energy consumption whose decrease is generally a strong incentive to selfish behaviours. Let us take an example of how we use service similarity in order to measure a context similarity. We assume for example that both a cooperative key establishment service and a signature delegation service require the same level of resources capabilities (asymmetric cryptography operations). So that, receiving a report about a node performing one of these security services at around the same resource capabilities level can be used to evaluate the trust level of this node for performing the other security service.

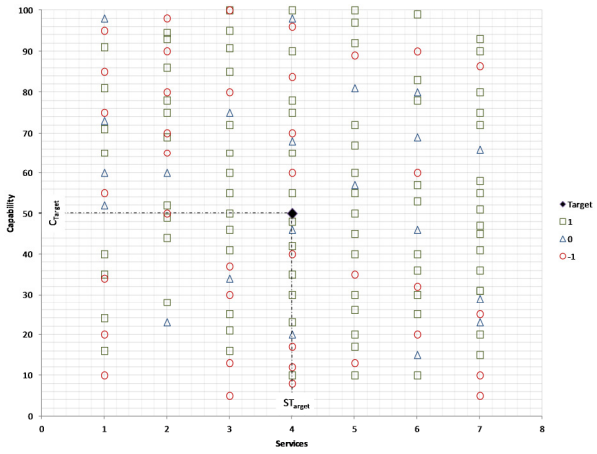


Fig. 24. Proxy reports history.

Figure 24 presents various reports R_{ij} (Service S_j, Capability C_j, Note N_j) stored at the trust manager, sent by all nodes j evaluating past interactions with a common assisting node P_i. This figure can be read as follows. The horizontal axis on the graph shows the different services for which the evaluated node P_i has provided assistance before. These services are ordered according to their resource-demanding requirements. The vertical axis shows the capabilities of the P_i node when assisting for these services. Each graph is characterized by the target report R_{Target} (S_{Target}, C_{Target}) depicted as a black diamond on figure 24:

- [S_{Target}] (*Service Target*) is the current service in request.
- [C_{Target}] (*Capability Target*) is the current P_i capability.

R_{Target} refers to the next report to be received, in case the proxy P_i is selected for the current service assistance. The goal of the context similarity computation process is to retrieve from the graph the most relevant reports, helping the trust manager to foresee the score received within the target report if the proxy P_i is retained for the service assistance.

Context similarity between a report about a previous interaction and the present target report is computed by considering a global contextual distance d_{ij} between the old report and the target one. To compute d_{ij}, we first define dS_j as the difference between the target service S_{Target} and the report service S_j and dC_j as the difference between the target capacity C_{Target} and the report capacity C_j.

$$dS_j = \left| S_{Target} - S_j \right| \tag{12}$$

$$dC_j = \left| C_{Target} - C_j \right| \tag{13}$$

We then obtain d_{ij} as:

$$d_{ij} = \min\left(\sqrt{(dS_{max}^2 + dC_{max}^2) \times \left(\frac{dS_j^2}{dS_{max}^2} + \frac{dC_j^2}{dC_{max}^2} \right)}, \sqrt{(dS_{max}^2 + dC_{max}^2) \times \left(\left(\frac{(S_{max} - S_j)}{S_{max} - (S_{Target} - \eta)} \right)^2 + \left(\frac{C_j}{C_{Target} + \eta} \right)^2 \right)} \right) \tag{14}$$

(for reports carrying a positive evaluation)

Or:

$$d_{ij} = \min\left(\sqrt{(dS_{max}^2 + dC_{max}^2) \times \left(\frac{dS_j^2}{dS_{max}^2} + \frac{dC_j^2}{dC_{max}^2} \right)}, \sqrt{(dS_{max}^2 + dC_{max}^2) \times \left(\left(\frac{(C_{max} - C_j)}{C_{max} - (C_{Target} - \eta)} \right)^2 + \left(\frac{S_j}{S_{Target} + \eta} \right)^2 \right)} \right) \tag{15}$$

(for reports carrying a negative evaluation)

The purpose of this computation is to make the distance metric more subtle than if it was merely measuring the sole similarity of an old report to a current situation. Indeed, some reports are meaningful although they are not close to the (S_{Target}, C_{Target}) target on the graph. To that respect, an asymmetry arises. A node behaving well for an expensive service is likely to behave well for a less demanding service too, whereas the fact that a node behaves well for a simple service gives no information about its expected quality when providing assistance for a demanding service.

The computation of d_{ij} takes this asymmetry into account by decreasing the distance (hence, increasing the probability to be selected) for the reports that give a good score when the evaluated node was at a much

lower capability level, or those that give a bad score when the evaluated node was at a much higher capability level. Figure 25 below explains how equations (3) and (4) orient the selection of the most relevant reports, and how the chosen parameters affect the distance computation. Indeed, each of these equations is obtained as the min of two terms. The first term merely relates to the distance between the evaluated report and the target. It is equal to $\sqrt{(dS_{max}^2 + dC_{max}^2)}$ for the points that belong to the ((S_{Target}, C_{Target}), dS_{max}, dC_{max}) ellipse, and tends to zero when a report gets closer to the center of that ellipse. The dS_{max} and dC_{max} , respectively x and y semi-axes of the ellipse, express the tolerance of the selection mechanism. The larger dS_{max} (resp. dC_{max}), the smaller the increase of distance when S_j (resp. C_j) gets further from S_{Target} (resp. C_{Target}).

The second term is where said asymmetry comes into play: it is proportional to the distance between the evaluated report and the point (S_{max}, 0) for positive scores, or to the distance between the evaluated report and the point (0, C_{max}) for negative scores. S_{max} refers to the most complex service in terms of resource consumption and C_{max} is the maximum resource level that could be available at a node. A positive report close to (S_{max}, 0) means that the candidate node performed well for a complex service while having only few available resources. A negative report close to (0, C_{max}) means that it performed poorly for a simple service, while being nevertheless at the maximum of its resources availability.

The parameter η is an adjustable parameter that allows to take into account through the second term a greater number of significant reports, by enlarging the upper-left and lower-right quarters of ellipses, thereby increasing the number of considered reports.

Finally, the computed d_{ij} distance is used as follows: a retained report R_{ij} should have a distance d_{ij} (R_{ij}, R_{Target}) < t, with t=$\sqrt{dS_{max}^2 + dC_{max}^2}$ acting as an adjustable threshold, characterizing the similarity interval we want to use.

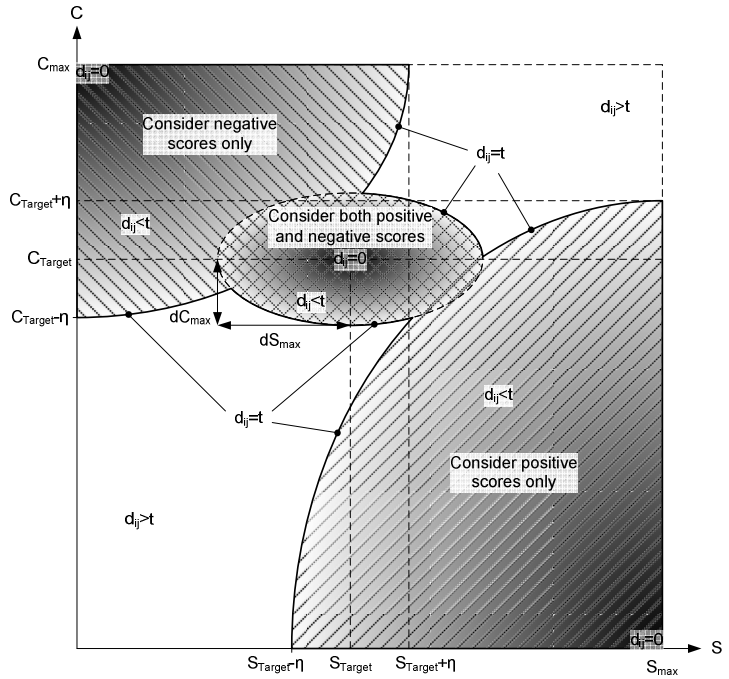


Fig. 25: Schematic representation of reports selection functions.

Three domains are represented on figure 25. The central ellipse is where reports are considered relevant under the dS_{max}, dC_{max} tolerance factors. The upper-left and lower-right quarters of ellipses, whose size can be adjusted through the η parameter, represent the areas where reports are meaningful in accordance with the score they carry. For each domain, the darker the shading colour, the lower the d_{ij} distance. The white areas represent the portions of the graph where the reports are not selected, since their computed distance exceeds the threshold t.

An example of the d_{ij} variation with (S_j, C_j) positive reports for (S_{Target}, C_{Target}, dS_{max}, dC_{max}) = (50, 70, 25, 15) is provided in figure 26.

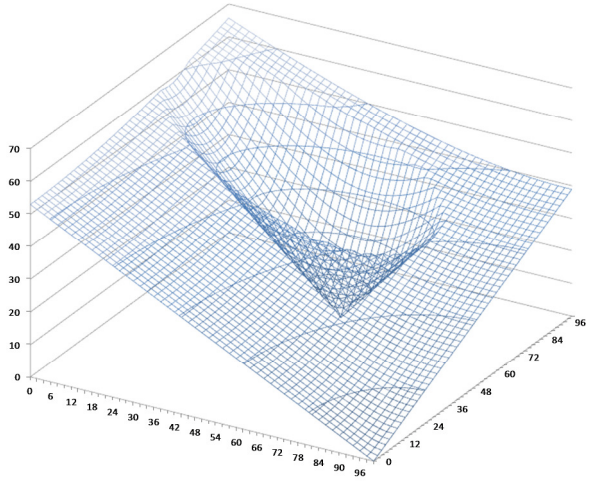


Fig. 26. Contextual distance for positive reports. As can be seen, the reports having the minimal d_{ij} distances are those that are either close to the target (central ellipse), or that reflect a node behaving particularly well in difficult conditions (bottom right corner).

Using the defined distance for restricting the set of considered reports leads to selecting only a subset of them, as shown below in figure 27.

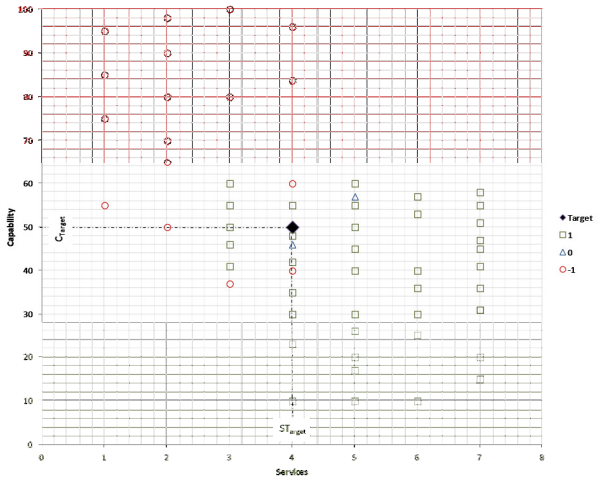


Fig. 27. Retained proxy reports. After computing the contextual distance of proxy P_i reports originally depicted in figure 24, only those for which $d_{ij} < t$ are retained.

Step 3: Computation of the weights $w_{R_{ij}}$ for each retained report R_{ij} in the step 2

Among the set of selected reports, not all have the same significance: those exhibiting a smaller contextual distance d_{ij} are more relevant than those with higher d_{ij} values. Meanwhile, old reports may not always be relevant for the ongoing trust rating, because a node may change its behaviour over time: recent reports are thus more meaningful than reports obtained for a long time. It is therefore necessary to assign a weighted value for each report, which bases on those two considerations and expresses the overall report relevance for the selection phase.

The weight w_{Rij} of the R_{ij} report is thus calculated as a product of two exponential factors that respectively decrease with the report age $(t_{now} - t_j)$ and the report contextual distance calculated above in step 2. The adopted scheme gives progressively less weight to older and contextually more distant reports.

$$w_{R_{ij}} = \lambda^{d_{ij}} \theta^{(s+1)(t_{now} - t_j)} \tag{16}$$

With

- λ, θ being parameters in the range of [0, 1] that express the 'memory' of the system. θ (resp. λ) is adjusted according to the expected rapidity of change in the observed node along the time (resp. among services). The lower θ (resp. λ), the lower importance the system gives to past (resp. contextually more distant) reports.
- $s = {}^{1}/_{2} * (N_j{}^2 - N_j)$ being a parameter computed from N_j (the score given by the witness node in the report R_{ij}) such that s is equal to 1 when this score is equal to -1 and 0 when this score is either 0 or 1. This way, the weight of negative score is doubled as compared to that of neutral or positive scores. The goal of the weighting factor *s* will be clarified in what follows.

Step 4: Computation of the trust value T_i for each proxy P_i

At this stage, the system is able to combine all opinions about the evaluated proxy P_i. This happens through a weighted average where the trustworthiness T_i of the proxy P_i for the sought collaborative service is eventually obtained as follows:

$$T_i = \frac{1}{\sum_{j=1}^{n} w_{R_{ij}}} \times \sum_{j=1}^{n} (w_{R_{ij}} . QR_j . N_j) \tag{17}$$

With:

- QR_j (Quality of Recommendation of the node j having issued the report R_{ij} about the proxy P_i) is the trustworthiness score assigned to a witness node depending on the accuracy of its past reports. It ranges between -1 and 1, 1 representing a very trustworthy node and -1 a node reporting the opposite of the actual service quality.
- $w_{R_{ij}}$ is the weighting factor computed above in step 3.

Step 5: Provision of the best rated proxies P_i

Upon computing trust levels for all selected candidates, the trust manager responds to the requesting node by securely providing it with the list of the best rated nodes, in accordance with the sought collaborative service and the respective current statuses of the assessed proxies.

4.2.2.3. Transaction and evaluation

In order to perform its planned collaborative service, the client node relies on the list of assisting nodes obtained from the trust manager. At the end of the transaction, the client node is able to assess the offered service received from each assisting node and sends a report to the trust manager in which it either rewards (positive score) or punishes (negative score) the participating nodes. The technique carried out to assess the offered assistance depends on the type of the service. It could be either derived from the client node local observations or from feedbacks received from other peers involved in the collaborative process, such as neighbours or the destination node. In our collaborative key establishment scheme, local observations may consist in suspicious communications occurring between proxies during the key exchange execution. These may mean that a collusion attack is being set up among the contacted proxies. The second assessment means, namely feedbacks received from other peers, is more likely. At the end of the collaborative key establishment procedure, the remote server B provides the client node with the list of participating nodes and/or those participating to the key exchange but having sent bogus shares. In turn, the client node transmits this list to the trust management system.

It is then of high importance to deal with received reports in our trust model by adequately taking care of the credibility of the node providing it. This is what the next 'Learning' operation is about.

4.2.2.4. Learning

The learning phase of our proposed trust management system qualifies it as a cognitive process. This phase is what distinguishes a cognitive process from an adaptive one. Translated to security scenarios, this means that adaptive security consists in dynamically reacting to a change in the environment by applying new security policies while cognitive security introduces a learning step wherein an assessment of the enforced action is carried out, which will eventually modify the system behaviour, so that a different action

may be taken when the same situation occurs. Indeed, a cognitive process is classically [163] described as a cycle involving four steps namely observation, planning, action and learning.

These steps almost straightforwardly correspond to the phases proposed in our TMS, as depicted in table 19.

Table 19: A Cognitive Trust Management system.

Cognitive process terminology	Proposed TMS terminology
Observation	Information gathering
Planning	Entity selection
Action	Transaction
	Reward and punish
Learning	Learn

The proposed learning phase consists of two steps: quality of recommendation update step and reputation update step.

4.2.2.4.1. Update of witness nodes' qualities of recommendation

Having received a report evaluating an assisting node, the trust manager learns about its behaviour. The trust manager can then update the trustworthiness score of all nodes having already sent a report about the same proxy, in similar contextual conditions. The underlying idea is quite simple: a node having previously marked as 'bad' a proxy node that eventually received a 'good' score will be considered a poor recommender (irrespective of its trustworthiness with respect to assistance in collaborative service, if any) and its Quality of Recommendation (*QR*) will be decreased (made closer to -1). Likewise, a node having previously given a good mark to a good-rated node will be considered as a good recommender, and its *QR* will be increased (made closer to 1).

This can be achieved by applying a weighted average function for trustworthiness score for each cooperative node on each node having sent a usable report about this cooperative node. This weighted average function serves two purposes. First, it avoids excessive variations of *QR*. For example, a generally good recommender will not suddenly be classified as a poor one for having issued a wrong report, but its recent history will mitigate its *QR* decrease. Second, the weighted average function allows to choose precisely to which extent a node's *QR* must be oriented either towards 1 (good recommender), or 0 (reporting non-usable data), or -1 (maliciously reporting the opposite of what happened). To that respect, weighting is important since a node being wrong in one old report relative to a contextually distant service will be far less penalized than a node being wrong in a very recent report about the same service, provided at the same capability level. The *QR* of the node having issued the report used to update the *QR*s of nodes having sent reports about the same proxy node is also an important parameter to take into consideration in the computation of the weight: saying the opposite of a very good recommender is more penalizing than contradicting a barely trustworthy recommender.

Let X be a witness node that helped the trust manager to evaluate a node P_i, which was used later as a proxy for assisting the node F. Depending on whether it has successfully accomplished the assigned task, the node F sends a report R_F to the trust manager that contains an evaluation score N: {-1: bad; 0: neutral; 1: good}.

The trust manager uses this report to update the recommendation trustworthiness score *QR* of each node having participated as a recommender during the proxy selection stage (which means that the report issued by this node must have been judged relevant at step 2, from contextual distance point of view). We defined X as being one such node.

The steps the learning stage is made up of are the following:

- First the system retrieves the *n* stored quality recommendation scores (that is, the history of their recommendation quality) for all witness nodes. X has for example QR^X $(QR_1, \dots, QR_{n-1}, QR_n)$ with QR_1 being the last updated (the most recent) quality recommendation score.
- The system then extracts the score N from the received report R_F and retrieves the weight w_{R_X} corresponding to R_X .

- Afterwards, it calculates QR_F, which represents the direction towards which the QR should evolve. QR_F is computed as follows:

$$QR_F^X = C_F \times r \quad \begin{cases} r = -|N^X - N^F| + 1 \\ C_F = w_{R_X} \times QR_1^F \end{cases} \tag{18}$$

In the above formula, r is computed from N^X (the score previously given by X) and N^F (the score just given by F) such that r is equal to 1 when these grades are identical, to -1 when they are opposite and to 0 when they differ by 1. r is therefore the value towards which the weighted average function should lean the QR of the witness node X, since this latter must tend towards 1 when the report is coherent with the newly received one, and tend to -1 when it contradicts it.
In accordance with our weighted average approach, C_F is the weight of r. As explained above, C_F increases when the weight of the report previously sent by X is high (an error by X is less tolerable if it pertains to a similar context). It also increases if F is a good recommender, as expressed with the QR_1^F factor (the current recommendation quality of node F).

- Finally, the system computes the new recommendation quality N_QR for node X as follows:

$$N\_QR^X = \frac{1}{\sum_{i=1}^{n} c_i \quad + \quad |C_F|} \times \left(\sum_{i=1}^{n} c_i \,.\, QR_i + QR_F^X \right) \tag{19}$$

The last term of this weighted average, r (in the form of QR_F^X that includes its weighting factor) has already been discussed above. The other terms are the QR_i, which are representative of the history of X's recommendation quality. Their respective weightings, c_i, are computed such as to be weighting values that assign a higher weight to the latest recommendation quality values. We propose to have c_i defined as (θ being presented in Step 3):

$$c_i = \theta^{(t_{QR_1} - t_{QRi})} \tag{20}$$

Once computed, the N_QR value is added to the QR historic list stored in the trust and reputation manager. It will be used as a recommendation quality for the future processes.

N_QR can fall off below zero and become negative, which means that the witness node is reporting the opposite of the real service quality. At that time, instead of applying a report discard, we propose to consider the opposite of what is provided in order to still make use of the maliciously reversed reports.

4.2.2.4.2. Update of assisting nodes' reputation levels

As explained above, we distinguish in this thesis between trust and reputation concepts. While *trust* measures the ability of a node to fulfil a specific task in a specific context, *reputation* refers to the global opinion of a node's trustworthiness in the network after having provided assistance for various services. The reputation level of an assisting proxy P_i is computed as follows:

$$Rep_{Pi} = \left(\sum_{j=1}^{n} c_j \,.\, N^{F_j} . QR_{F_j} \right) \tag{21}$$

N^{Fj} is the score given by the requesting node F_j having obtained the assistance from P_i for a specific service and QR_{F_j} is its quality of recommendation. The weighting factor c_j, presented above, is applied to gradually forget old feedbacks.

It is important to update reputation levels of nodes in the network after each interaction in order to identify assisting nodes commonly judged as untrustworthy. Upon receiving a feedback from the requester node F,

the trust manager takes into account its evaluations to recalculate the reputation levels of the involved assisting nodes. If the reputation level of one of these falls below a certain threshold, its activity is interrupted and it is added to a list of ill-reputed nodes. It is also reported by the trust manager to the network operator, which may then examine the reasons for its misbehaviour. Indeed, a node might provide wrong information or bad services either due to a deliberate, malicious misbehaviour, or just as a result of a malfunction or an environmental change.

4.2.3. Synthesis

In this subsection, we provide a quick assessment of how the proposed solution as well as the most common prior art trust models behave when matched against the design decisions that were identified in previous chapter.

Table 20: Assessment of the proposed solution and the most common trust management systems against the identified best practices.

	CONFIDANT [151]	CORE [153]	Zhu Han et al. [158]	Proposed solution
Functional trust decisions	routing services	multi-service	cooperative radio transmissions	multi-service
Contextual trust decisions	not addressed	not addressed	not addressed	compute context similarity to gather the most significant reports and derive trust scores
Trust scores computation	single global score for routing service	single global score for multiple services	single global score for radio transmission service	• take into account variable node status and assigns dynamic trust scores for each service assistance and node capabilities. • define a second trust score reflecting the recommendation quality of a node reports
Reports trustworthiness evaluation	checks the global trust level of the witness node to evaluate the credibility of the corresponding report	evaluating trustworthiness of reports is not addressed	all reports are trusted as long as the reporter is never classified as a liar node	check the recommendation quality score of the witness node to evaluate the credibility of the corresponding report
Exchanged observations	only negative observations are exchanged in reports	only positive observations are exchanged in reports	both positive and negative observations are exchanged	both positive and negative observations are exchanged
Reports weighting factors	no weighting factors are assigned	gives more weight to past reports	no weighting factors are assigned	• give more weight to recent and context-similar reports • give more weight to negative observations • give more weight to reports provided from nodes with high recommendation quality scores
Storage and decisions making localisation	• local observations and other nodes reports are stored at the node • local trust decisions making	• local observations and other nodes reports are stored at the node • local trust decisions making	• local observations and other nodes reports are stored at the node • local trust decisions making	• local observations are reported to a centralized entity which provides nodes with trust decisions making on demand
Learning about decisions making	once an assisting node trust level falls below a threshold, it is excluded from future routing path selections	once an assisting node trust level falls below a threshold, all of its service requests are denied and it may only act as a service provider	• detect lying nodes and send them warning message • once the number of received warning messages exceeds a threshold the witness node is discarded • no punishment decisions concerning misbehaving assisting relays are specified	• update the recommendation quality of previously involved witness nodes • once the recommendation quality of a witness node approaches to -1 the system considers the opposite of what is provided • once an assisting node trust level falls below a threshold, it is excluded from the future selections an all its service requests are denied

4.3. TRUST MODEL TECHNICAL IMPLEMENTATION

4.3.1. TMS subsystems overview

The trust model we propose can be viewed as a package of functionalities linked with one another in order to ensure a reliable trust decision and offer the best assistance to the requesting node.

As shown in the following figure (Fig. 28), the proposed TMS consists of various subsystems with different roles and functionalities. There are three main components, namely the database, the core and the input/output interface.

- The *Database (DB)* is a structured collection of useful information gathered from the environment;
- The *Core* is the smart component of the system performing functions such as analyse, computation and update;
- The *Input/output interface* is the interface used to communicate and exchange information with the requesting nodes.

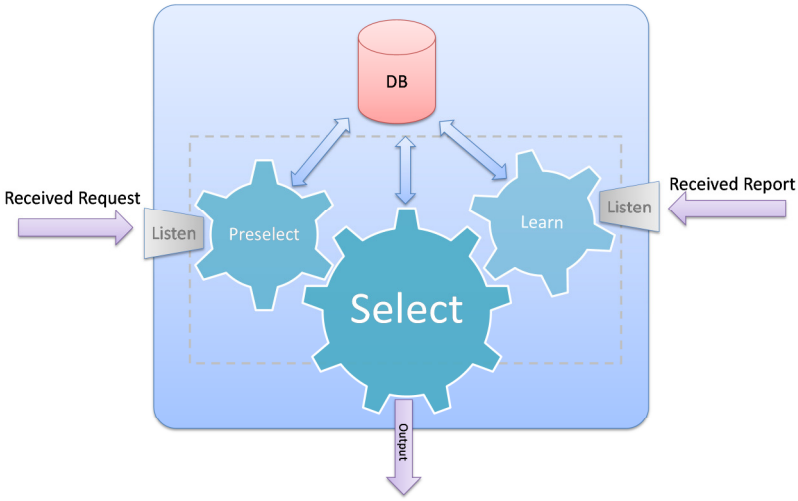


Fig. 28. Proposed trust management system.

4.3.2. TMS subsystems design

Among the three components our proposed TMS is made up of, two can be qualified as major ones and will be discussed in the rest of this subsection: these are the core of the system and the database. We present in the following the structure and the role of each of these components.

4.3.2.1. Database design

Reliability and robustness of the proposed system rely on the quantity and quality of stored data, since computing a node trust level requires the knowledge of its past behaviours. To that aim, the database component saves all information that will be helpful in the decision-making.

We designed the TMS database in two steps, namely conceptual and logical modelling.

- **Conceptual modelling** allows to model data at higher level, learning about the different involved entities and how they relate to one another;
- **Logical modelling** derives from the conceptual modelling and presents the final appearance of the database.

Based on the trust model specifications, we extracted the following constraints in order to define attributes and relationships for the Entity-Relationship diagram corresponding to the proposed TMS:

- Network topology contains one or many nodes;
- Each node has a particular conduct (fair behaviour or misbehaviour);
- Nodes must share secrets with the neighbourhood;
- Each node belongs to one or more group, for example multicast or neighbouring groups;

- Each node has a type (proxy node, able to provide collaborative services and/or simple node, able to consume collaborative services);
- Each node can execute one or more collaborative service(s) (e.g. routing, aggregation, signing-verification, encryption-decryption, key establishment);
- The TMS must keep all QR values stored in the database;
- The TMS must process each request sent by a node;
- The TMS must respond all nodes requests by assigning one or many assistant node(s).

Figure 29 represents the logical model of our system. It contains seven main entities, namely:

- **Node:** to store all nodes that make up the system
- **Node Type:** to store the different types of nodes that exist in the system
- **Service:** to store the different existing services in the system
- **Group:** to store the group(s) within which the nodes of the studied topology fall
- **Misbehaviour:** to store the intrinsic nodes behaviours
- **Quality_Recom**: to store the Quality of Recommendation score of the node
- **Trust req**: to store the exchanged request

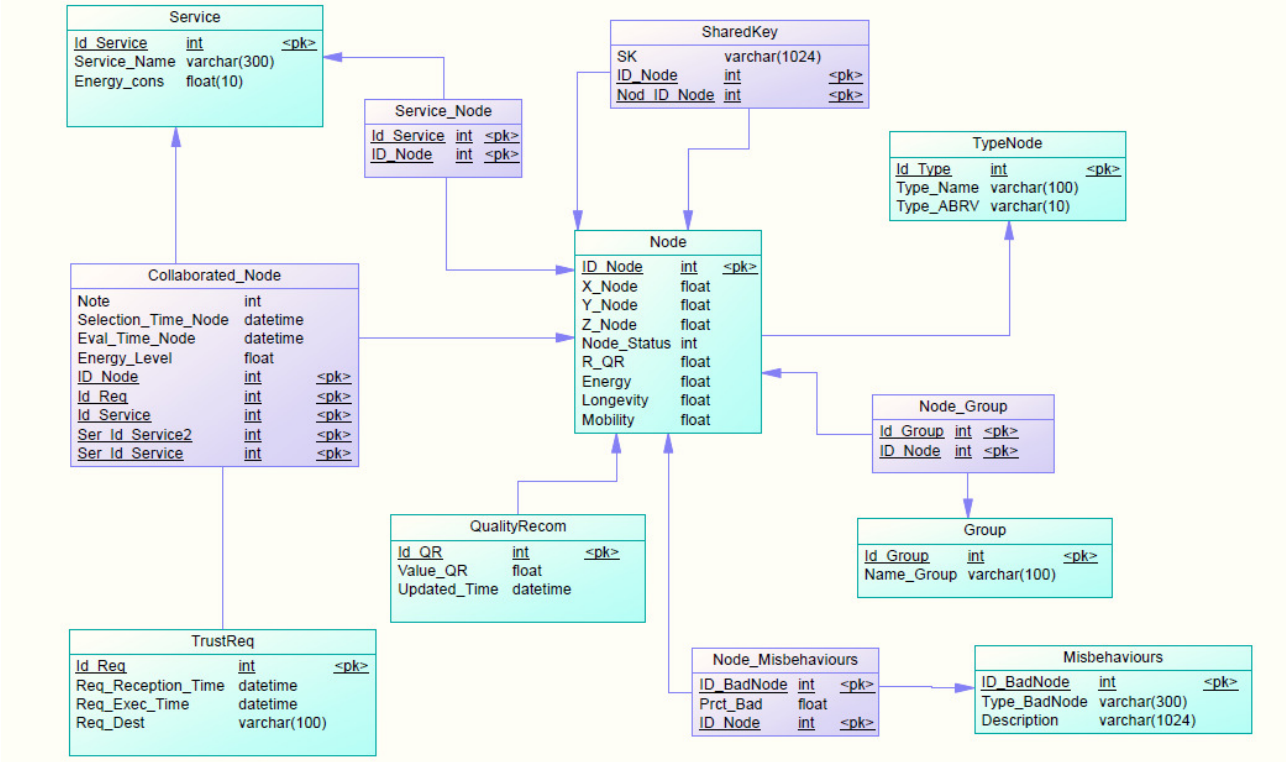


Fig. 29. Logical model of TMS Database

4.3.2.2. Core design

The core design relies on a modular approach, wherein the core component is seen as being made of multiple "blocks", or logical entities, as depicted on figure 30. These blocks interact and communicate with each other. We present in what follows the different building blocks of our trust model, along with their specifications and algorithmic solutions.

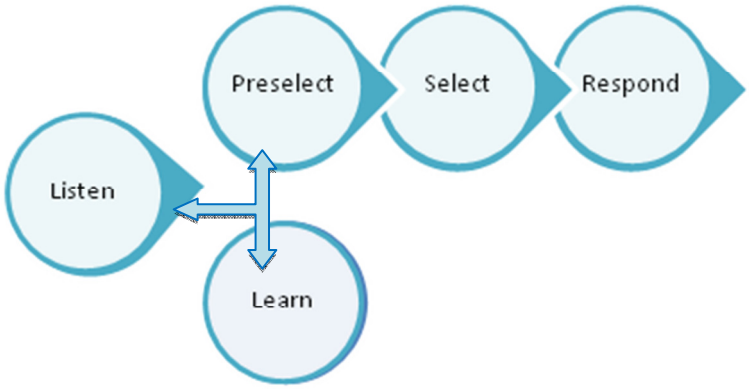


Fig. 30. TMS blocks.

4.3.2.2.1. Listen block

The Listen block is the first block in our TMS. It is responsible for listening to any node's request. This block can be designed as a server model that enables a client node to establish a connection in order for this latter to send and receive information from the TMS, in the form of a request for assistance and the associated proxies list response.

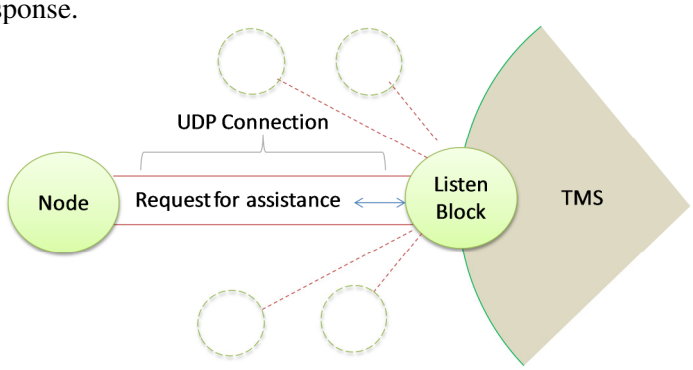


Fig. 31. Listen block.

The connection between the two communicating entities is ensured by UDP sockets. Indeed, the UDP transport protocol is more suitable than the heavier TCP for what concerns IoT nodes.

4.3.2.2.2. Preselect block

The main goal of the Preselect block is to increase the relevance of the decision task in the next step, through the narrowing of all assisting nodes into a subset of most relevant candidates (N in the figure 32), which can be able to assist the requesting node.

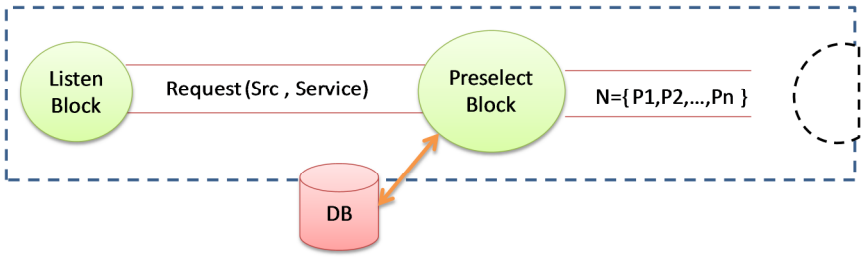


Fig. 32. Preselect block.

At the receipt of the request, the Listen block forwards it to the Preselect block. At this stage, the Preselect block consults the TMS database and restrains the set of potential candidates. This preselection depends on both the service requirements and the proximity link (for what concerns collaborative key establishment, nodes have to share a secret key with the requester, and they may have to belong to the same group).

Algorithmically, we can present the Preselect block as a function called FnPreselect(), described as follows:

- *IN_Pnode*: contains the entire set of proxy nodes
- *OUT_Pnode*: will contain the subset of preselected nodes
- *ReqSRC*: the source requester

```
function FnPreselect (ReqSrc, IN_Pnode, OUT_Pnode) {
        for each Pi ∈ IN_Pnode do
        check whether Pi is able to perform the requested service
                if (True) then
                        check its suitability to assist ReqSrc
                        [Has a shared secret and belongs to the same group]
                        if (True) then
                                add Pi to OUT_Pnode
}
```

At the end of this phase, the Preselect block is able to deliver the list of selected proxies to the Select block in order for this latter to analyse it and determine the preselected proxies trust levels, with respect to the requested service.

4.3.2.2.3. Select block

The Select block is the main engine of the proposed trust management system. It is responsible for the trust decision making and implements most of the computational operations described above. The main goal of this block is to assess the trust level of each proxy P_i belonging to the subset of nodes received from the Preselect block.

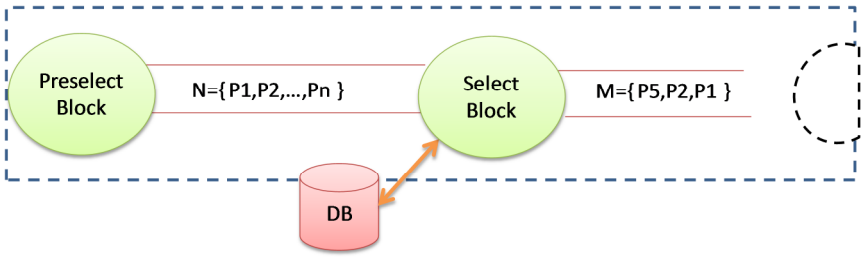


Fig. 33. Select block.

The Select block collects each stored report that is related to the proxy P_i, selects the most relevant ones and computes each report weight.

Algorithmically, we can present the Select block as a function called FnAct(), described as follows:

- *IN_Pnode*: input table containing the nodes retained from the Preselect block
- *OUT_Pnode*: output table including the best-rated nodes, assigned to the requesting node to assist its collaborative service
- *Report*: local table storing reports related to the current proxy

```
function FnAct (IN_Pnode ,OUT_Pnode) {
        for each Pi ∈ IN_Pnode do {
                Get_Report (Pi, Report);
                SumT ← 0;  SumCoeff ← 0;
                For each Rj ∈ Report do {
                        QRj ← Quality of recommendation score related to the report Rj originator
                        NRj ← Score evaluating Pi given in the report Rj
                        Wj ← Calcul_weight (Rj);
                        SumT ← SumT + QRj * NRj * Wj;
```

$SumCoeff \leftarrow SumCoeff + Wj;\}$

$TrustPi \leftarrow (\frac{1}{SumCoeff}) * SumT$

}

Add the best rated proxies to the *OUT_Pnode* table

}

Get_Report (*Pi, Report*): This function evaluates the stored reports about P_i and selects the most relevant ones. As explained above, a retained report must have a contextual distance lesser than a threshold *t*.

Calcul_Weight (*Report*): This function computes the weight for each report passed in parameter and returns a weighting score as a float value.

4.3.2.2.4. Respond block

Upon computing trust levels for all selected proxies in the Select block, the trust manager responds to the requesting node by providing it with the list of the best-rated nodes. This operation is ensured by the Respond block.

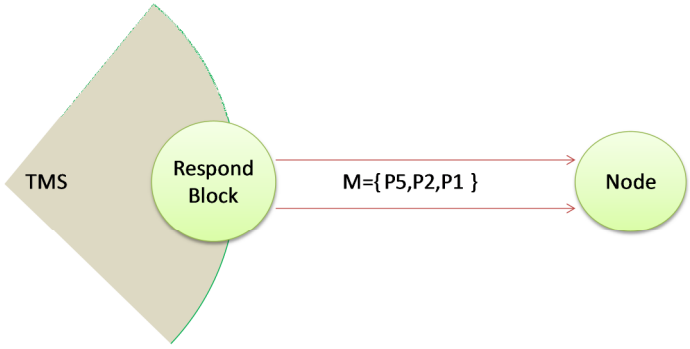


Fig. 34. Respond block.

4.3.2.2.5. Learn block

In order to perform its service on a collaborative basis, the requesting node relies on the list of assisting nodes obtained from the trust manager. After the service completion, the requesting node is able to assess the service obtained from each assisting node and sends a report enclosing this assessment to the TMS.

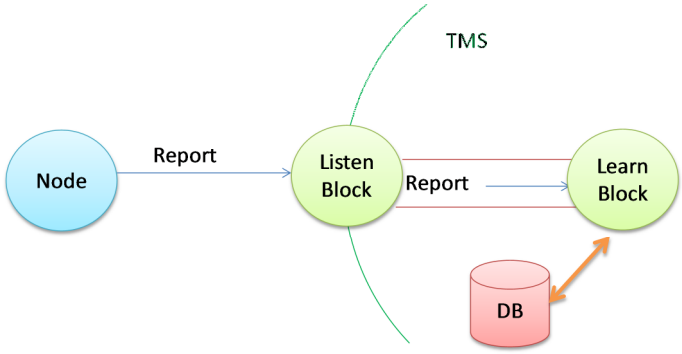


Fig. 35. Learn block.

The reception of this report triggers an update function. This function is instantiated within the Learn block, which analyses the received reports in order to update the QRs of witness nodes as well as the reputation of assisting nodes.

Algorithmically, we can present the Learn block as a function called FnLearn() described as follows:

- *IN_Report*: Received report

function FnLearn *(IN_Report) {*

```
for each Pi ∈ IN_Report do {
    Ni ← Note related to Pi
    Fill the W vector with the witness nodes that helped TMS in selecting Pi

        for each Wi ∈ W do {
            Fill the QR vector with previous QR scores related to the witness node Wi

            NewQR ← Comp_NQR (QR, Ni, Pi )

            add the NewQR value to the database
        }
    }
}
```

The main role of the Comp_NQR() function is to compute the new quality of recommendation of the witness node that helped the TMS to choose P_i as an assistant node.

4.3.2.3. State diagram

The transitions between blocks were controlled following the state diagram shown on the graph below (figure 36).

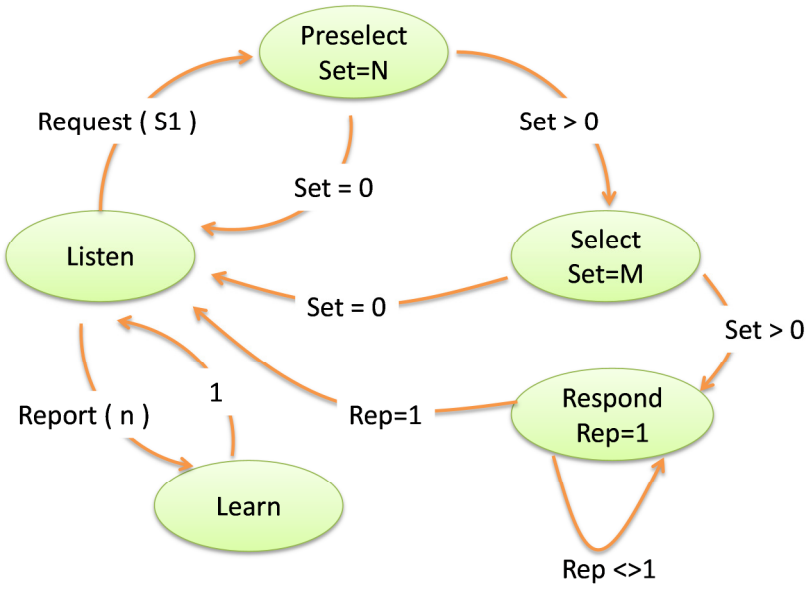


Fig. 36. TMS state diagram.

Once started, the system turns to the Listen state. At this stage, the system will wait until a message is received from a client on the input interface. Received messages can be either requests or reports. If it is a request, the system switches to the Preselect state; otherwise, it moves to the Learn state.

The system then continues its evolution in accordance with the type of received data. Indeed, in case the system is in to the Preselect state, it performs all preselection functions described above and returns a value, called Set, representing the potential proxies selected at this stage. If Set is null then the system reverts to the Listen state. Otherwise it switches to the Select state where it first eliminates the less relevant reports and then calculates the trust level of each of these potential candidates obtained while in the Preselect state. Finally, it provides the list of the best-rated proxies to the requesting node in the Respond state.

In case the system is in the Learn state, it performs the update functions and then reverts to the Listen state.

4.4. SIMULATION AND PERFORMANCE ANALYSIS

In this section, we provide performance results in order to prove the proper operation and effectiveness of the proposed system. These results were obtained through the development of a dedicated simulation framework, which was favoured over the use of an existing networking simulation environment, such as OmNet++ [164] or NS [165]) for efficiency and simplicity reasons. Indeed, our simulation framework makes it easy to implement specific design decisions (e.g. databases customisation, trust patterns, behaviours and interactions model), as well as to integrate and add new functionalities while, in the meantime, allowing for straightforward porting onto actual physical devices. Also, graphical outputs were conceived so as to meet our specific requirements.

4.4.1. Simulation lifecycle

The operational phases of our simulator are depicted in figure 37 below and explained later in this subsection.

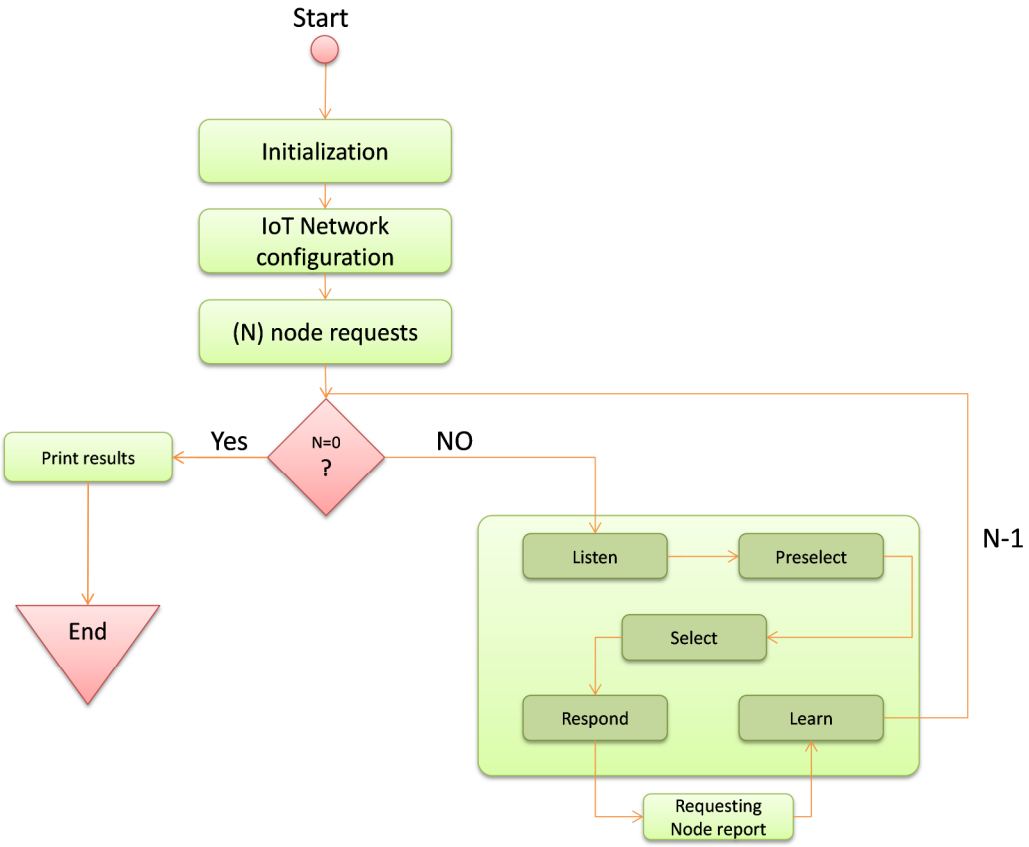


Fig. 37. TMS operational phases.

4.4.1.1. Initialisation

During the initialisation phase, the simulator clears all stored data within the database. To do so, the system calls the database package and executes the InitiateDatabase() function for each database table.

4.4.1.2. IoT Network configuration

This phase is where the initial network configuration takes place. The system defines the network topology according to the configuration parameters (number of proxy nodes, number of requesting nodes, proportion of poor witness nodes, percentage of malicious nodes, initial qualities of recommendation, groups

and services). It generates a set of nodes with random attributes (e.g. services, group, (x,y,z) position, real quality of recommendation) and maintains them in the database. These attributes will be used during the simulation lifecycle.

For the rest of this chapter, we consider the following configuration (table 21):

Table 21: Simulation configuration parameters.

Number of proxy nodes (PNs)	100
Number of nodes	200
Poor witness nodes (%)	20%
Malicious nodes (%)	10%
Initial quality of recommendation (*QR*)	1
Services	6

Once the connection with the database is established, the simulator executes the InitiateDatabase() function in order to ensure that the database is empty. At this stage, the network topology generation can be launched.

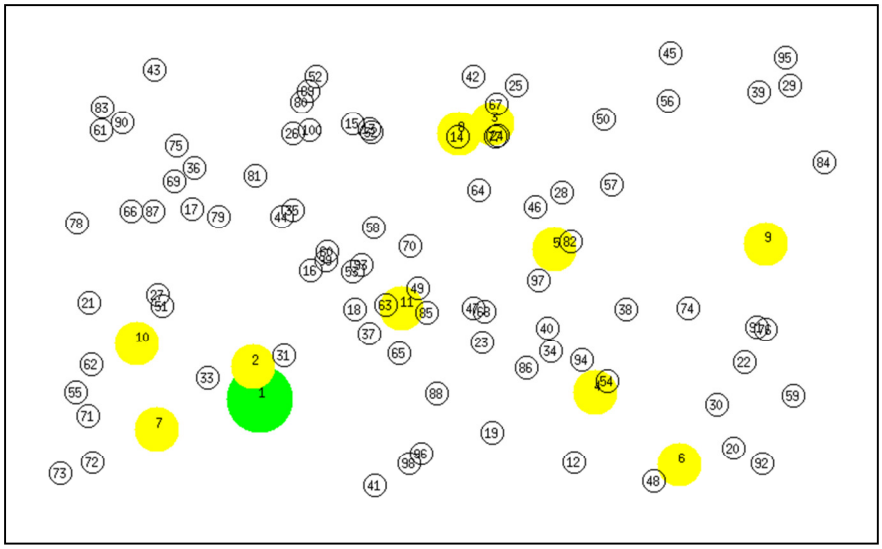


Fig. 38. Generated IoT network topology.

As an example, figure 38 represents a topology where our simulator generated 111 nodes:

- 10 proxies (*yellow colour*);
- 100 constrained nodes (*empty/white colour*);
- 1 trusted entity (including the TMS) (*green colour*).

Each of the simulated nodes is characterized by a set of attributes, e.g. for node 12:

Table 22: Simulated node attribute set.

Node ID	*12*
Position (x,y,z)	*(600,20,0)*
Quality of recommendation (QR)	*1*
Services	*{S4,S2}*
Malicious node	*False*
Real quality of recommendation (R_QR)	*0.8*

This node is located at location (600, 20, 0). It is able to run the services S4 and S2. From the system point of view, its quality of recommendation is initially set to 1 in order to be adjusted progressively revealing its real trustworthiness level as a witness node. Its real quality of recommendation is set to 0.8 in this example and is of course not known by the trust manager.

At the beginning of the lifetime of the network, all nodes are assumed to be trustworthy and well-behaving since they are supposed to be verified for failures before deployment. Once the network becomes operational, it may happen that nodes become compromised. Their trustworthiness levels can therefore change with respect to their behaviours.

4.4.1.3. Request/Response simulation

Once the topology is defined, the simulator activates all nodes (figure 39).

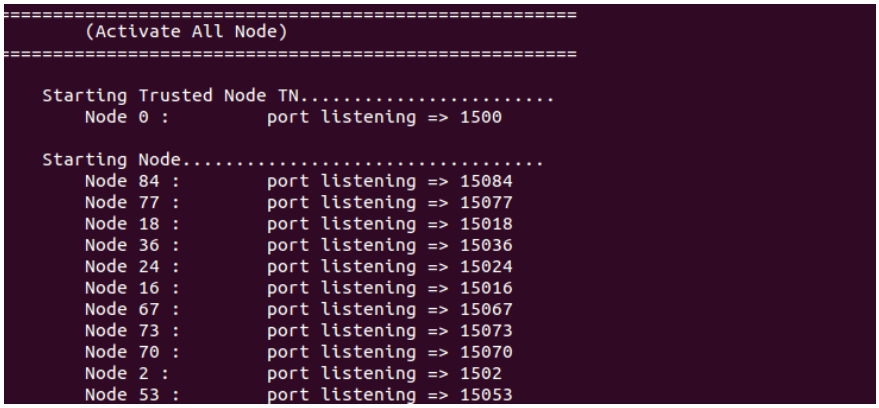


Fig. 39. Activated nodes and their respective listening ports.

Once the network becomes operational, the simulator selects a random node, generates a request for assistance and sends it to the trust manager. Based on the type of service and specific requirements of the requesting node, the system selects potential proxies that run this service and fulfil its requirements.

This pre-selection phase is performed within the Preselect state, as explained in the previous section. Figure 40 shows the system response upon receiving a request from the node 12, requesting assistance for a key establishment service. Proxies 1, 2, 3, 4, 7, 8, 9 and 10 are able to assist this key establishment service, and proxies 1, 3, 4, 5, 9 and 10 share a key with the node 12. Hence, only proxies 1, 3, 4, 9 and 10 are retained for the subsequent selection step.

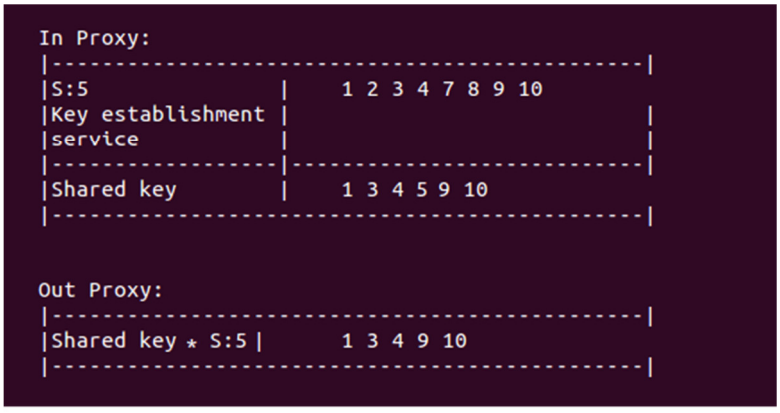


Fig. 40. Preselect state results.

At this stage, the trust manager switches to the trust decision making process. Based on a set of selected reports, the TMS computes trust levels of preselected proxies 1, 3, 4, 9 and 10. The figure 41 below presents the results of this phase. To obtain proxy 1 trust level, 332 reports have been analysed.

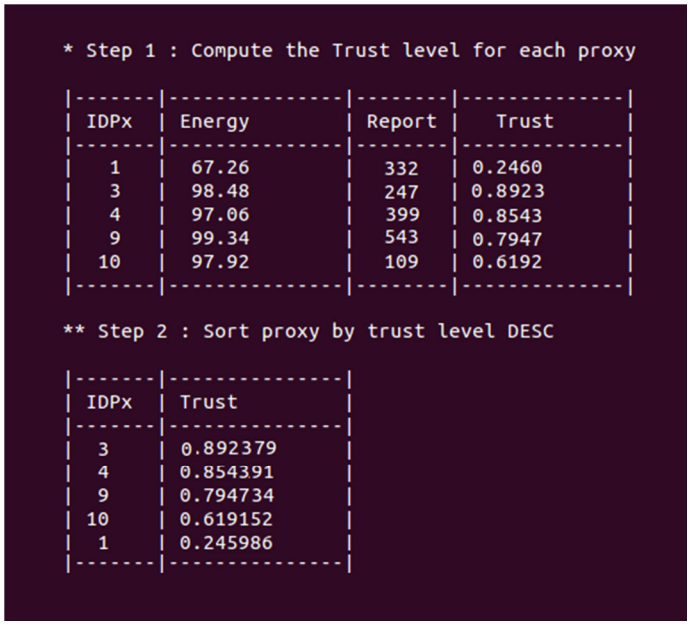


Fig. 41. Select state results.

Upon computing trust levels for the proxies 1, 3, 4, 9 and 10, the TMS responds to the node 12 request and provides it with the list of the best rated proxies. The node 12 will rely on a set of nodes that contains proxies 3, 4, 9 and 10 (we assume that the key establishment service requires the involvement of 4 collaborating proxies).

After the service execution, the node 12 is able to assess the service obtained from each proxy and sends a report assessment to the TMS (figure 42). The evaluation score depends on the node assessment. This score is randomly generated, according to the malicious status of the assisting node and to the real quality of recommendation of the requesting node (fixed at the start-up stage).

Fig. 42. Respond state and received reports.

Having received a report evaluating proxy 4, 9, 3 and 10, the TMS learns about their respective behaviours and updates the *QR* of all witness nodes having already sent a report about the same proxies (figure 43).

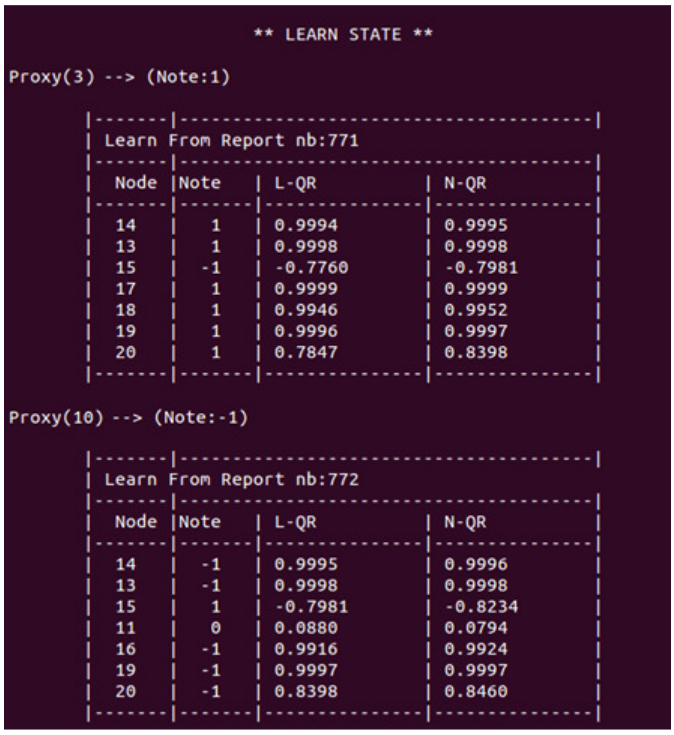


Fig. 43. Learn state results.

The scores "1" and "-1"given by node 12 to respectively evaluate the proxy 3 and 10 induce the decrease of the *QR*s of witness nodes having previously assigned different scores while they increase the *QR*s of those that had previously given identical scores. This variation of witness nodes *QR*s is adjusted basing on the *QR* of the node 12 itself and on other weighting parameters, as described in previous section.

4.4.2. Performance evaluation

4.4.2.1. Evoluation of quality of recommendation score

As mentioned in the design of our solution, a real quality of recommendation *R_QR* is set, that defines the intrinsic behaviour of each node when reporting evaluations about other nodes. These subsequent evaluation reports are then used as input in the trust manager in order to calculate trust values. A clear vision of the quality of recommendation influences thus directly trust computations, leading to reliable decision makings and offering the best assistance to requesting nodes: discarding poor/lying recommenders and promoting efficient recommending nodes are indeed required in order to have the computed trust match the actual trustworthiness of an assisting node.

This provides us with a simple means to check whether the proposed TMS behaves properly: if yes, the interpolated quality of recommendation should tend towards the real quality of recommendation. The figures below show examples of the evolution of the quality of recommendation for some nodes.

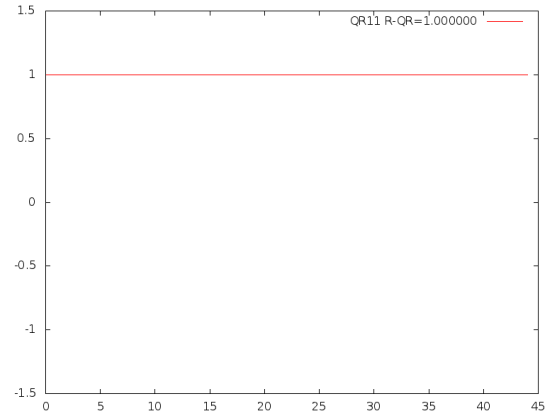


Fig. 44. A perfect recommender (*QR*=1) is recognized as such by the trust manager, which constantly assigns it the "1" score as quality of recommendation. No incident interferes with this rating.

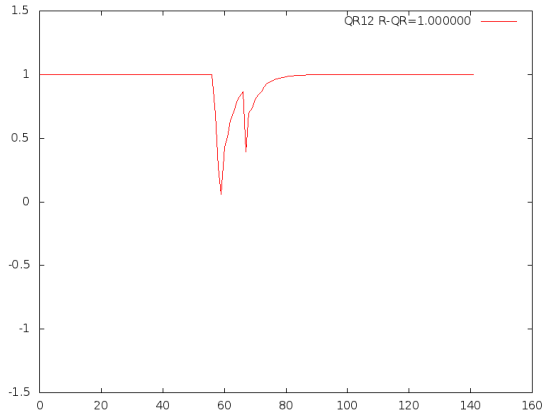


Fig. 45. Perfect recommender and poor witnesses. Here, a node that is intrinsically a perfect recommender has its quality of recommendation score initialized at 1. Two incidents, caused by poor witnesses' errors cause the trust manager to decrease its *QR* score. However, the system behaves properly and quickly reverts to the proper value.

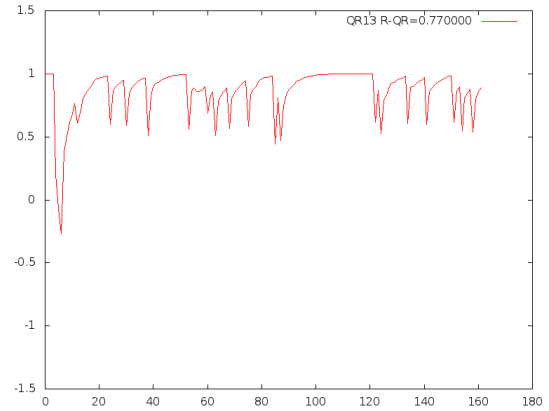


Fig. 46. Good recommender. We see here a situation where the considered node is a good, yet not perfect, recommender. *QR*=0.77 means that the node, though generally giving a good recommendation, will be wrong 23% of time. Hence, as compared with the previous case, this node's quality of recommendation is not only affected by poor witnesses but also by its own errors.

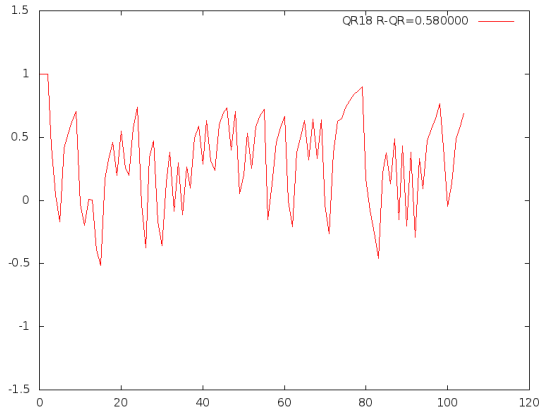


Fig. 47. Poor recommender. With an even lower real *QR* (*QR*=0.58), the node's score is regularly affected by its own mistakes, in addition to the wrong reports from poor witnesses. The score therefore oscillates between 0 (node is estimated to issue useless reports) and 1 (node is estimated to issue trustworthy reports), with rare occurrence of negative scores (node is estimated to be intentionally issuing false reports). Mitigation of these oscillations would require relying on non-linear formulas: trying to mask them with slower increase/decrease slopes only would also slow down the convergence of the system for recognizing a fully trustworthy node (more frequent case) and would therefore damage the entire system behaviour.

Simulation results confirm the proper operation of the proposed trust management system. As shown above, positions of curves in the vicinity of the *R_QR* prove that the integrated learning module performs properly and succeeds in fine-tuning the quality of recommendations.

4.4.2.2. Detection of misbehaving assisting nodes

Based on the received reports evaluating each proxy and the recommendation quality of their originating nodes, we can detect assisting nodes with bad reputations. As shown in the figure 48 below, the proxy 10 had more than 60% of bad evaluations. Fixing the threshold of bad evaluations to 60%, our trust management system considers it as a misbehaving node.

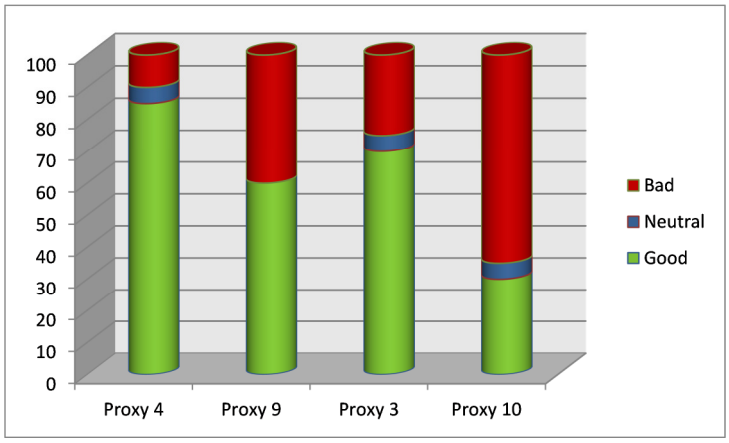


Fig. 48. Assessment of proxies 4, 9, 3, 10 reputations.

4.4.2.3. Protection against attacks

Previously, we have proven the proper operation of our TMS when making trust decisions, fine-tuning *QR*s of witness nodes and detecting misbehaving assisting nodes. Meanwhile, a trust model which is built to counter internal attacks inside a collaborative group and reduce the impact of misbehaving nodes can be itself hindered by specific attacks that can disrupt its functioning. It is especially vulnerable to three potential attacks that have been classified as critical in [166]. In the following, we investigate the effectiveness of our TMS under these three threats, namely bad mouthing / ballot stuffing, selective misbehaviour, and on-off attacks.

4.4.2.3.1. Bad mouthing and Ballot stuffing attacks

As long as reports from witness nodes are taken into account in a trust management system, the risk of receiving wrong recommendations is present.

Malicious nodes may provide dishonest recommendations either to boost the trust values of malicious accomplices (referred to as the *ballot stuffing* attack) or to drop trustworthiness of honest parties (referred to as the *bad mouthing* attack). Currently, there is no trust management system in wireless communications that can deal with these two types of attacks without making initial assumptions about the behaviour of the nodes. The CONFIDANT trust model allows only negative reports to be propagated, thereby assuming that bad mouthing attacks could not be performed by a node. As for the CORE model, collected reports take into account positive reports only, thereby assuming that a node has no advantage to carry out ballot stuffing attacks for unknown nodes benefit. Other trust management systems either do not address these attacks and consider all reported evidences as reliable, or are content with checking the global reputation of a node to weigh its reports.

The trust management system we propose in this thesis defends against these attacks by building and updating separately trust recommendation values from regular trust values. Our trust management system involves a learning phase allowing it to learn from the consequences of its actions in the entity selection phase. This knowledge is used to fine-tune the trustworthiness of previously used recommendations, in order to improve the selection in the future. As presented above, Quality of recommendation scores (*QR*s) are computed by checking consistency between the current evaluation and previous recommendations used during the proxy selection phase. A malicious witness node can

be detected during the learning process by putting its dishonest recommendations up against others' evaluations, which progressively decreases its *QR* and reduces the impact of its recommendations during the entity selection phase.

As shown in figure 49, without considering the *QR*, the trust level of an honest node (red graph) significantly drops when impacted by a bad mouthing attack (characterized in this case by a group of ten witness nodes sending negative evaluations about a well-behaving assistant node). Considering the *QR* of a node when assessing its reports, the system becomes able to decrease the *QR* of these malicious witness nodes by putting their dishonest recommendations up against others' evaluations. Our trust model (blue graph) decreases in a first time the node trust level but quickly recovers its trustworthiness by reducing the impact of wrong reports provided by malicious nodes.

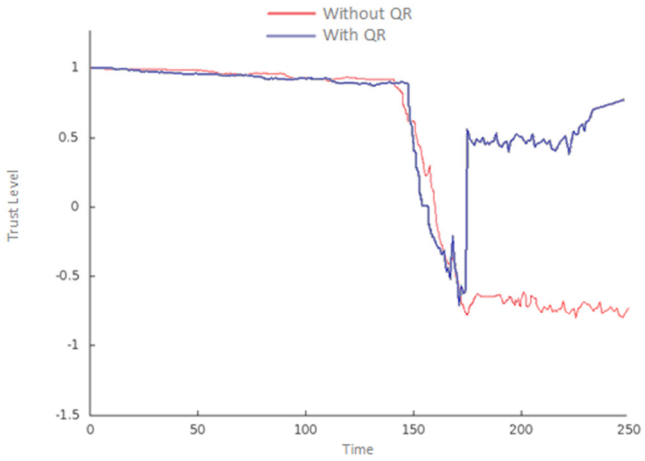


Fig. 49. Resilience against bad mouthing attack.

4.4.2.3.2. On-off attack

This attack exploits the forgetting property of trust management systems, which gives more weight to recent recommendations. Such weight adjustment is required since trust is variable over time. For example, in wireless communications, a honest mobile entity may suffer for a period of time from bad channel conditions, which deteriorate its trust level as a relay node. After it moves to a location where the channel condition is better, it should be made possible for that node to recover its original trust level. Hence, old and recent recommendations about a node do not carry the same weight.

However, a dishonest entity can take advantage of this property and behave alternatively well and badly, since it can compensate past bad behaviours by behaving well for a period of time and eventually regaining trust. This attitude is referred to as an on-off attack.

In order to make our trust model robust against such attacks, we adapt our system such that a bad behaviour will be memorized for a longer time than a good behaviour. We accordingly add a weighting factor *s* in the computation of the report age in step 2, so that we make negative scores appear less old, compared with neutral and positive nodes.

This decision discourages dishonest nodes to recurrently switch between bad and good behaviours and require them to perform many good actions to recover their trust values.

We see in figure 50 a situation where the node changes its behaviour alternatively. It behaves well for the ten first interactions. Then it provides bad services for the second ten interactions and reverts to normal. Without considering *s* (blue graph), the system takes more time to detect the bad behaviour of the node since the node past good behaviour is more emphasized, and therefore hides the malicious transition for longer. Once the system recognizes this bad behaviour and starts to slightly decrease its trust level, the node stops bad behaviours and regains trust. With the use of *s* (red graph), the system detects earlier the node misbehaviour and decreases its trust level. Since bad behaviours are memorized for a longer time, it takes much longer for the node to regain trust from the system point of view.

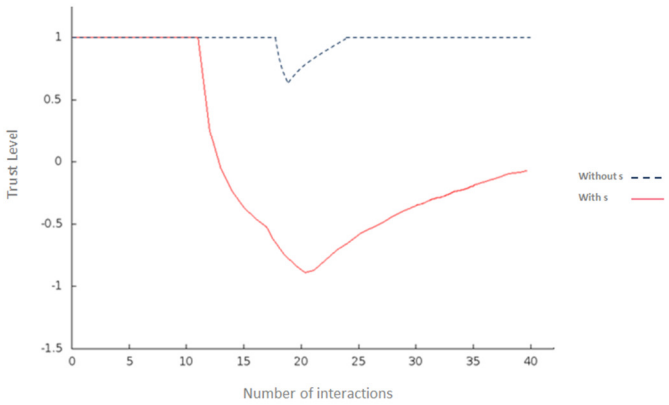


Fig. 50. Resilience against on-off attack.

4.4.2.3.3. Selective behaviour attack

While a dishonest node switches between bad and good behaviours over time in the on-off attack described above, it can also behave alternatively badly and well between services. This attack is referred to as selective behaviour attack. If a node behaves well for simple services, it can still behave badly for other resource-demanding services. Thereby, the average trust level will remain positive and the node will selfishly save energy.

Existing trust models suffer from this attack since they rely on a unique trust value that globally characterizes a node including all assisted services. Our system defends against selective behaviour attack through the implementation of a functional model that assigns multiple trust values to a node, in relation with all assisted services. A node that would always perform poorly in demanding collaborative services would always receive bad scores, which would not be compensated for by good scores obtained for good behaviour in simpler services. In the short term, this means that the node carrying out this attack would no longer be selected for demanding services, which it would no longer be in position to damage. In the longer term, such behaviour could trigger action from the system administrator, if the accumulation of poor scores reaches a predetermined threshold.

We consider in figure 51 a situation where the trust level of a dishonest node is evaluated with respect to a resource demanding service. We can see that this node, being considered under a global trust value, manages to hide its misbehaviour when performing this service. It maintains an overall high trust level (red graph) since it compensates received bad scores with good scores obtained for its good behaviours in simpler services. Our trust model (blue graph) succeeds to decrease the trust level of the node when performing this specific service despite its good behaviours in other services.

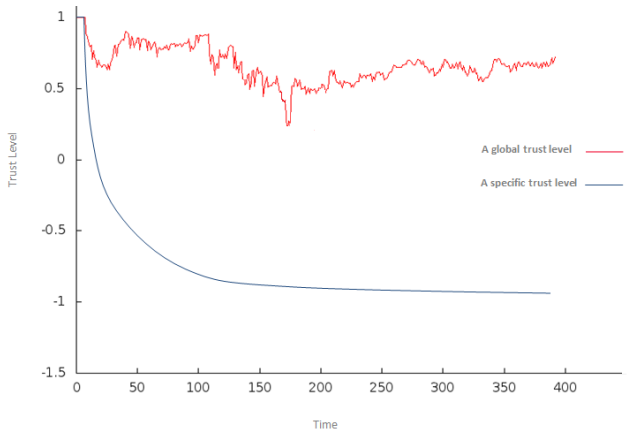


Fig. 51. Resilience against selective behaviour attack.

4.5. CONCLUSION

In this chapter, we proposed a multi-service and context-aware trust management system as required for our collaborative key establishment approaches and, more generally, as required for collaborative networking services. Indeed, this trust model manages cooperation and enables nodes requesting for assistance to identify the best partners when setting up collaborative networking services. The proposed trust model fulfils the specific requirements of the environment we consider, characterized with heterogeneity and nodes energy constraints. At the same time, it goes beyond the shortcomings identified through our study of prior trust models, such as flexibility to handle variations of nodes and/or services contexts and ability to process all reported information without making initial assumptions about the behaviour of nodes. A qualitative comparison of TMSs is provided in this chapter to show the strengths of our proposal, as compared with prior trust models.

In addition, a set of testbeds and simulation results have been reviewed in order to prove the proper operation and effectiveness of the proposed system. This effectiveness is assessed by considering how our trust management system responds to specific situations. Among these situations was its ability to fine-tune the real quality of recommendation of a witness node through its reported evidences, its efficacy to identify a misbehaving proxy and take true decisions and its conduct when subjected to specific attacks that may be launched against trust management systems.

CONCLUSION

This thesis addresses new security issues in the Internet of Things (IoT). The heterogeneous nature of IoT communications, coupling resource-constrained networks with powerful Internet makes it challenging to provide end-to-end secured communications between IoT entities. Indeed, applying existing end-to-end key establishment protocols with their heavy resource demands could be hindering for most IoT components due to their low capabilities in terms of computing power and energy resources. Since the IoT will not emerge through the design of entirely novel protocols, these security standards have to be revisited in order to adapt them to the IoT scenarios. In that light, this thesis provides several significant contributions aiming at addressing IoT security challenges and specific requirements. Each contribution was presented and detailed in a separate chapter.

CHAPTERS SUMMARY

The first chapter is a thorough overview of existing end-to-end security standards and key establishment schemes in the literature and a study of the generic design decisions helping to characterize a key establishment protocol for the Internet of Things. Indeed, we provided a classification of key establishment protocols according to three criteria: the key delivery scheme (key transport or key agreement), the underlying cryptographic primitive family (symmetric or asymmetric) and the authentication method. Considering the initial requirements of the IoT, we have retained TLS Handshake, Internet Key Exchange and HIP BEX protocols as the best candidates for key establishment in the IoT. However, when assessing them in terms of energy efficiency, we have highlighted their resource-intensive design. Then we gave an in-depth study of the efficient key establishment solutions that have been proposed for constrained devices, from legacy WSNs to Internet-integrated pre-IoT topologies.

The second chapter proposes novel collaborative approaches for key establishment designed to moderate the requirements of existing security protocols, in order to be supported by resource-constrained devices. Contrary to prior proposals, we explored the possibility of reducing the computational load to be performed on constrained devices through collaborative offload instead of doing so by relying on weaker cryptographic algorithms. Our solution exploits spatial and temporal heterogeneity of nodes in the Internet of Things to offload heavy computational load required at the constrained device to more powerful nodes in the surroundings. Retained TLS Handshake, Internet Key Exchange and HIP BEX protocols are redesigned so that the constrained communicating party may delegate its expensive cryptographic operations to less constrained nodes. During the key exchange, these assisting nodes take charge of the session key derivation, in a collaborative and distributed manner. Two distributed techniques have been proposed and carefully designed to perform the collaborative key exchange approach. The first distributed approach depends on reliable multiple hop-by-hop deliveries of secret fragments by proxies (dedicated assisting servers). In case these proxies are non-dedicated nodes belonging to the same local infrastructure of the constrained device – though being less impacted by energy constraints – misbehaving and/or unavailability behaviours may arise. In order to reinforce the reliability of the collaborative approach, a second threshold distributed technique is proposed enabling the recovery of the session key at the two endpoints of the communication even in case of proxies misbehaviour or unreliability. A formal security analysis performed using AVISPA tool has validated the security of our collaborative variants of TLS Handshake, Internet Key Exchange and HIP BEX. Assessed from the points of view of cryptographic and communication costs, our proxy-based schemes show a significant gain in terms of energy at the constrained device compared with the basic approaches of key establishment standards.

The third chapter is an exhaustive overview of the literature in the field of collaborative networking services. We highlighted the vulnerability of these emerging collaborative approaches to internal attacks, launched from within the group of cooperating nodes, that may prevent the system from working properly. We then assessed the security mechanisms proposed in the literature to counter these attacks. We classified these mechanisms into two main categories: security-by-design mechanisms and trust-based mechanisms. Much attention was especially devoted to studying existing trust models and identifying a set of relevant design practices to use as part of a generic trust management system, required to ensure the proper operation of our proposed collaborative key establishment services.

The fourth chapter focuses on the design of a new trust management system that fulfils the requirements of our collaborative approaches and bypasses identified shortcomings of prior trust models. This trust model manages cooperation between nodes and enables them to identify the best partners when setting up collaborative networking services. It takes into account variable node status and assigns dynamic trust scores for each class of service assistance and node capabilities. It also handles received reports from witness nodes without any initial restrictions, basing on their quality of recommendation scores. These scores are updated during a learning phase and kept independent from the scores evaluating them as assistants in a collaborative service. In order to evaluate the performance of the proposed trust model, we have developed our own experimental simulation environment. The system performance was assessed by considering different aspects. Simulation results proved the proper operation of different integrated modules and formulas in our trust management system. Obtained graphs proved that the integrated learning module succeeds in fine-tuning the quality of recommendations. Revealing the real trustworthiness level of a witness node makes our trust model able to take relevant decisions since trust is built based on reported evidences from previous experiences. We also proved its effectiveness against potential attacks targeting trust models namely bad mouthing, selective misbehaviour, and on-off attacks. Obtained results showed that the system recognizes quickly malicious attempts trying to induce these attacks and succeeds to overcome them before they affect the proper operation of the trust model.

DISCUSSION AND OPEN ISSUES

There are many interesting open issues that deserve further investigation:

- Specifying the number of proxies for our collaborative key establishment schemes: Selecting the right number of proxies is not an easy task. Obviously, we cannot specify the exact number of them without other joint parameters. Indeed, this number should be a function of the network size and topology, the degree of resilience required against attacks and the quantity of resources that a proxy is devoting to collaborative services. That is, it would be interesting to carry out simulations to identify the appropriate number of proxies according to the variation of these parameters. It is evident that choosing a small number of proxies causes bottleneck and creates performance problems while selecting a high number of proxies increases the communication and, in certain cases, computational overhead during the protocol exchange.
- Making the proxy-based approach transparent at the server side: allowing this transparency makes it possible for the constrained node to take assistance from proxies and delegate to them its heavy cryptographic operations while the server remains unaware of this phase during the key establishment process. In that case, the proposed proxy-based solution will require new protocol implementations at the constrained device only, which make our approach more flexible. However this solution increases the computational charge at the constrained device since this latter gets involved in more transactions with the proxies and more computations.
- Protecting our collaborative approach against collusion attacks: while most studies only consider attacks coming from an individual node, the assumption that a group of malicious nodes may collude is often overlooked despite its probability of occurrence in collaborative services. Collusion attacks are even more detrimental and hard to detect than individual

attacks since a group of nodes may coordinate to achieve a common malicious purpose. In our collaborative key establishment schemes, assisting nodes can collude by gathering their private fragments to recover the session key between the source and the destination. This type of collusion attack occurs undetectably since the system still works properly during the session key exchange. However it will have serious impacts later on, when a secure communication using this "secret" key will start between the source and the destination. This attack has been considered for the design of our trust model. We assume that the constrained node would be likely to be able to detect communications between assisting nodes during the key exchange as long as they are within the same radio range. Yet, malicious proxies may postpone the collusion attack once the key exchange has been completed, in order to make sure that the constrained node is no longer monitoring their activities. As a first way to defeat collaboration of malicious nodes during the supporting mechanism, the constrained node may keep a small key fragment of the premaster secret (of a size equivalent to the final session key) that it would transmit later to the server, encrypted with the server public key. For a small fragment (as opposed to the entire premaster secret), the encryption overhead on the constrained node would remain limited. Further work in this direction would be interesting in order to fully grasp the possibilities of nodes collusions in collaborative services and take security measures (further enhancements of our trust model) against these threats.

- Studying the situation where the two endpoints of the communication are resource-constrained nodes. It might be gain incentive to check whether the two peers can rely on the same set of assisting nodes at the same time, or if two distinct groups of proxies have to be assigned.

Let us conclude this thesis with another open issue which is not specifically related to the present study but rather to the general field of key establishment in the IoT.

- Give more interest to Lamport and Merkle tree signatures. Lamport signatures are proposed to be used by proxies in our solution to perform signatures replacing heavier asymmetric algorithms while Merkle scheme is used as a binary tree for authentication of Lamport signature verification keys. These two schemes make it possible to create digital signatures based on one-time signature schemes. With the advent of quantum computing, widely-used signature schemes such as RSA, DSA and ECC are threatened and about to become entirely insecure, whereas the former two schemes relying on hash functions are conjectured to be unbreakable using quantum computers. It would be promising to investigate the use of one-time signature schemes along with symmetric ciphers, also resistant to quantum computing attacks, for designing quantum-safe cryptosystems. Especially, to investigate the adaptability of such cryptosystems to constrained devices. Likely, memory capacity should be the most hindering factor for these.

THESIS PUBLICATIONS

JOURNAL

- Y. Ben Saied, A. Olivereau, M. Laurent and D. Zeghlache, *Lightweight collaborative keying for the Internet of Things*, submitted to Elsevier Ad hoc Networks, 2013.
- Y. Ben Saied, A. Olivereau, D. Zeghlache and M. Laurent, *A Survey of Collaborative Services in Modern Wireless Communications and their Security-related Issues*, submitted to Elsevier Journal of Network and Computer Applications, 2013.
- Y. Ben Saied, A. Olivereau, D. Zeghlache and M. Laurent, *Trust Management System Design for the Internet of Things: A Context-Aware and Multi-Service Approach*, submitted to Journal of Computer Security, 2013.

INTERNATIONAL CONFERENCES

- Y. Ben Saied, A. Olivereau and D. Zeghlache, *Energy Efficiency in M2M Networks: A Cooperative Key Establishment System*, 3rd International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT) 2011.
- Y. Ben Saied, A. Olivereau and D. Zeghlache, *Etablissement de clé de session en environnement M2M entre nœuds à ressources fortement hétérogènes*, Computer & Electronics Security Applications Rendez-vous (C&ESAR) 2011.
- Y. Ben Saied and A. Olivereau, *HIP Tiny Exchange (TEX): A Distributed Key Exchange Scheme for HIP-based Internet of Things*, 3rd International Conference on Communications and Networking (ComNet) 2012.
- Y. Ben Saied, A. Olivereau and M. Laurent, *A Distributed Approach for Secure M2M Communications*, 5th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2012.
- Y. Ben Saied and A. Olivereau, *D-HIP: A Distributed Key Exchange Scheme for HIP-based Internet of Things*, First IEEE WoWMoM Workshop on the Internet of Things: Smart Objects and Services (IoT-SoS) 2012.
- Y. Ben Saied and A. Olivereau, *(k, n) Threshold Distributed Key Exchange for HIP based Internet of Things*, 10th ACM International Symposium on Mobility Management and Wireless Access (MOBIWAC) 2012.
- Yosra Ben Saied, Alexis Olivereau and Radhouene Azzabi, *COACH: a COntext Aware and multi-service trust model for Cooperation management in Heteregenous wireless networks*, 9th International Wireless Communications and Mobile Computing Conference (IWCMC) 2013.

PATENTS

- Y. Ben Saied, A. Olivereau and C. Janneteau, Système distribué d'établissement de clé de session pour nœud à faibles ressources, 2012.
- Y. Ben Saied, A. Olivereau and C. Janneteau, Méthode et système d'établissement d'une clé de session, 2012.

EUROPEAN PROJECT DELIVERABLE

- Internet of Things Architecture (IoT-A) Project Deliverable D4.2, Concepts and Solutions for Privacy and Security in the Resolution Infrastructure, February 2012.

- Internet of Things Architecture (IoT-A) Project Deliverable D3.3, Initial IOTP Protocol Suite Definition, April 2012.
- Internet of Things Architecture (IoT-A) Project Deliverable D4.4, Final Design and Implementation Report, to be published in May 2013.
- Internet of Things Architecture (IoT-A) Project Deliverable D3.6, IOTP Protocol Suite Definition, to be published in May 2013.

REFERENCES

[1] F.L. Lewis, Wireless sensor networks, in: D.J. Cook, S.K. Das (Eds.), Smart Environments: Technology, Protocols, and Applications, Wiley, 2004.

[2] W. Geng, S.Talwar, K. Johnsson, N. Himayat, K.D. Johnson, "M2M: From mobile to embedded internet," Communications Magazine, IEEE, vol.49, no.4, pp.36-43, April 2011.

[3] C. Wietfeld, H. Georg, S. Groening, C. Lewandowski, C. Mueller, J. Schmutzler, "Wireless M2M Communication Networks for Smart Grid Applications," Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), April 2011.

[4] Y. Zhang, R. Yu, S. Xie, W. Yao, Y. Xiao, M.Guizani, "Home M2M networks: Architectures, standards, and QoS improvement," Communications Magazine, IEEE , vol.49, no.4, pp.44-52, April 2011.

[5] A. J. Menezes , S. A. Vanstone , P. C. Van Oorschot, Handbook of Applied Cryptography, CRC Press, Inc., Boca Raton, FL, 1996.

[6] W. Diffie and M.E. Hellman, New directions in cryptography, IEEE transactions on information theory 22, 644-654,1976.

[7] J. Arkko, E. Carrara, F. Lindholm, M. Naslund and K. Norrman, MIKEY: Multimedia Internet KEYing , IETF RFC 3830, August 2004.

[8] V. Manral, Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH), IETF RFC 4835, April 2007.

[9] T. Dierks, E. Rescorla , The Transport Layer Security (TLS) Protocol Version 1.2, IETF RFC 5246, August 2008.

[10] E. Rescorla, N. Modadugu, Datagram Transport Layer Security, IETF RFC 4347, April 2006.

[11] P. Eronen, H. Tschofenig, Pre-Shared Key Ciphersuites for Transport Layer Security (TLS), IETF RFC 4279, Decemeber 2005.

[12] C. Kaufman, Internet Key Exchange (IKEv2) Protocol, IETF RFC 4306, December 2005.

[13] Moskowitz, R., Nikander, P., Jokela, P., and T. Henderson, Host Identity Protocol, IETF RFC 5201, April 2008.

[14] D. Zhang, X. Xu, J. Yao, Z. Cao, Investigation in HIP Proxies, draft-irtf-hiprg-proxies-04 (IRTF work in progress), October 2011.

[15] R. Moskowitz, Host Identity Protocol Architecture, draft-ietf-hip-rfc4423-bis-03 (IETF work in progress), September 2011.

[16] P. Urien, "HIP support for RFID," draft-urien-hip-tag-03 (IETF work in progress), December 2009.

[17] TelosB datasheet. Available: http://moss.csc.ncsu.edu/~mueller/rt/rt11/readings/projects/g4/datasheet.pdf

[18] 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, Washington, DC, USA, November 18-22 2002, pp. 41–47.

[19] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in IEEE Symposium on Security and Privacy, Berkeley, California, May 11-14 2003, pp. 197–213.

[20] D. Liu and P. Ning, "Location-based pairwise key establishments for static sensor networks," in Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks (CCS '03), pp. 72–82, October 2003.

[21] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in Proceedings of IEEE INFOCOM'04. 2004.

[22] S. Schmidt, H. Krahn, S. Fischer, D. Watjen, "A security architecture for mobile wireless sensor networks," in Proceedings of the 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS'04) (Heidelberg, Germany), Vol. 3313, August 2004.

[23] A. Perrig, R. Szewczyk, V. Wen, D. Cullar, and J. D. Tygar, "Spins: Security protocols for sensor networks," in Proceedings of the 7th Annual ACM/IEEE Internation Conference on Mobile Computing and Networking (MobiCom), Rome, Italy, July 2001, pp. 189–199.

[24] J. Lopez, "Unleashing public-key cryptography in wireless sensor networks," Journal of Computer Security, vol. 14, no. 5, pp. 469–482, 2006.

[25] N. R. Potlapally, S. Ravi, A. Raghunathan, N.K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols", IEEE Transactions on Mobile Computing, 128–143, 2006.

[26] D. J. Malan, M. Welsh, M. D. Smith, "A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography," First IEEE International Conference on Sensor and Ad Hoc Communications and Networks, 2004.

[27] P. K. Dutta, J. W. Hui, D. C. Chu, and D. E. Culler, Securing the deluge network programming system, in Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN). ACM, New York, NY, 326–333. 2006.

[28] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, P. Kruss, TinyPK: Securing sensor networks with public key technology, in Proceedings of the 2nd ACM workshop on Security of Ad Hoc and Sensor Networks, pp. 59–64, 2004, USA.

[29] R.L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM, vol. 21, no. 2, pp 120–126, February 1978.

[30] Q. Huang, J. Cukier, H. Kobayashi, B. Liu, J. Zhang, "Fast authenticated key establishment protocols for self-organizing sensor networks," in Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications, ACM Press, 2003; 141–150.

[31] P. Kotzanikolaou, E. Magkos, D. Vergados, and M. Stefanidakis, "Secure and practical key establishment for distributed sensor networks," in Security and Communication Networks, Wiley InterScience, 2009.

[32] W. Hu, P. Corke , W. C. Shih, L. Overs, secFleck: A Public Key Technology Platform for Wireless Sensor Networks, in Proceedings of the 6th European Conference on Wireless Sensor Networks, February 11-13, 2009, Cork, Ireland

[33] G. Gaubatz, J. Kaps, and B. Sunar, "Public key cryptography in sensor networks–revisited," Lecture Notes in Computer Science, vol. 3313, pp. 2–18, 2005.

[34] G. Gaubatz, J. Kaps, E. Ozturk, and B. Sunar, "State of the art in ultralow power public key cryptography for wireless sensor networks," in Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops. IEEE Computer Society Washington, DC, USA, 2005, pp. 146–150.

[35] J. Mache, C.-Y. Wan, and M. Yarvis, "Exploiting heterogeneity for sensor network security," in Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pp. 591–593, 2008.

[36] R. Riaz, A. Naureen, A. Akram, A. Akbar, K. Kim, and H. Farooq Ahmed, "A unified security framework with three key management schemes for wireless sensor networks," Computer Communications, vol. 31, no. 18, pp. 4269–4280, 2008.

[37] P. Eronen H. Tschofenig,"Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", IETF RFC 4279, December 2005.
[38] Vipul Gupta , Matthew Millard , Stephen Fung , Yu Zhu , Nils Gura , Hans Eberle , Sheueling Chang Shantz, Sizzle: A Standards-Based End-to-End Security Architecture for the Embedded Internet, Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications, p.247-256, March 08-12, 2005.
[39] ANSI X9.62. "Elliptic Curve Key Agreement and Key Transport Protocols". American Bankers Association,1999.
[40] ANSI X9.63. "The Elliptic Curve Digital Signature Algorithm". American Bankers Association, 1999.
[41] W.Jung et al (2009), "SSL-based Lightweight Security of IP-based Wireless Sensor Networks", International Conference on Advanced Information Networking and Applications Workshop.
[42] S. Raza, S. Duquennoy, T. Chung, D. Yazar, T. Voigt, and U. Roedig, "Securing Communication in 6LoWPAN with Compressed IPsec", In Proceedings of the 7th IEEE International Conference on Distributed Computing in Sensor Systems (IEEE DCOSS 2011), Barcelona, Spain, June 2011.
[43] V. Nagalakshmi, I. Rameshbabu and P.S. Avadhani," Modified protocols for internet key exchange (IKE) using public encryption key and signature keys. Proc. of the eighth international conference on Information Technology: New Generations 2011; 376-381.
[44] R. Sangram, G. P. Biswas, "Establishment of ECC-based Initial Secrecy Usable for IKE Implementation", in Lecture Notes in Engineering and Computer Science, pages 530-535, 2012.
[45] T. Aura, Cryptographically Generated Addresses (CGA), IETF RFC 3972, March 2005.
[46] R. Moskowitz, HIP Diet EXchange (DEX), draft-moskowitz-hip-rg-dex-05 (IETF work in progress), March 2011.
[47] T. Heer, LHIP Lightweight Authentication Extension for HIP, draft-heer-hip-lhip-00 (IETF work in progress), February 2007.
[48] N. R. Potlapally, S. Ravi, A. Raghunathan, N.K. Jha, A study of the energy consumption characteristics of cryptographic algorithms and security protocols, IEEE Transactions on Mobile Computing, 128–143, 2006.
[49] A. Liu and P. Ning. TinyECC: A configurable library for Elliptic Curve Cryptography in Wireless Sensor Networks. Technical Report TR-2007-36, North Carolina State University, Department of Computer Science, 2007.
[50] G. De Meulenaer, F.Gosset , F.-X. Standaert, and O. Pereira, On the energy cost of communication and cryptography in wireless sensor networks, Proceedings of IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WIMOB 2008).
[51] L. Atzori, A. Iera, and G. Morabito, The Internet of Things: A Survey, Elsevier Computer Networks, 2010.
[52] M. Petrova et al, " Overall secure PN architecture," in My Personal Adaptive Global NET (MAGNET), D2.1.2/D4.1.3,Octobre 2005.
[53] R. Merkle, Secrecy, authentication, and public key systems, Ph.D. dissertation, Dept. of Electrical Engineering, Stanford Univ, 1979.
[54] N. Fazio, and A. Nicolosi, Cryptographic Accumulators: Definitions, Constructions and Applications. Technical report ,2002.
[55] L. Lamport, "Constructing digital signatures from one-way function," in Technical Report SRI-CLS-98, SRI international ,October 1979.
[56] S. Seys and B. Preneel, Power consumption evaluation of efficient digital signature schemes for low power devices, in Proceedings of the 2005 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (IEEE WiMob 2005), pages 79–86, 2005.
[57] M. Watson, Basic Forward Error Correction (FEC) Scheme, RFC 5445, March 2009.
[58] A. Brusilovsky, I. Faynberg and Z. Zeltsan, Password-Authenticated Key (PAK) Diffie-Hellman Exchange, RFC 5683, February 2010.
[59] W. Dai, Crypto++ Library 5.6.0, http://www.cryptopp.com.
[60] J. Lacan, V. Roca, J. Peltotalo, Reed-Solomon Forward Error Correction (FEC) Schemes, IETF RFC 5510, April 2009.
[61] IT++ Library, http://itpp.sourceforge.net/current/
[62] A. Wander, N. Gura, H. Eberle, V. Gupta, and S. C. Shantz, Energy analysis of public-key cryptography for wireless sensor networks,, Third IEEE International Conference on Pervasive Computing and Communications, , pages 324–328, 2005.
[63] G. De Meulenaer, F.Gosset , F.-X. Standaert, and O. Pereira, On the energy cost of communication and cryptography in wireless sensor networks, Proceedings of IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WIMOB 2008).
[64] J. Paek, K. Chintalapudi, R. Govindan, J. Caffrey, and S. Masri, A wireless sensor network for structural health monitoring: performance and experience, IEEE Computer Society Washington, DC, USA, 2005.
[65] C. Merlin and W. Heinzelman, Duty cycle control for low-power-listening mac protocols, in MASS, 2008.
[66] A. Armando et al, "The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications," in Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, Springer, Heidelberg (2005), http://www.avispa-project.org
[67] D. Dolev and A. C. Yao. On the security of. public key protocols. In Proc. 22th IEEE Symposium on Foundations of Computer Science, pages 350-357, 1981.
[68] Y. Glouche and T. Genet. "SPAN – a Security Protocol ANimator for AVISPA – User Manual," IRISA / Université de Rennes 1, 2006. 20 pages. http ://www.irisa.fr/lande/genet/span/.
[69] E. Royer and C.-K. Tob, " A Review of Current Routing Protocols for Ad Hoc Wireless Networks", IEEE Pers. Commun., pp.46 -55 1999
[70] C. Perkins, E. Royer, S. Das, RFC 3561 Ad hoc On-Demand distance vector (AODV) routing. 2003
[71] D. B. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. Proceedings of the Workshop on Mobile Computing Systems and Applications, pp. 158-163, IEEE Computer Society, Santa Cruz, CA, December 1994.
[72] A. Shamir, How to share a secret, Communications of the ACM, v.22 n.11, p.612-613, Nov. 1979
[73] G. R. Blakley, Safeguarding cryptographic keys, In Proc. Nat. Computer Conf. AFIPS Conf. Proc, vol.48, p. 313–317, 1979.
[74] X. Zhang and He, "Collusion Attack Resistance and Practice-Oriented Threshold Changeable Secret Sharing Schemes", AINA '10 Proceedings Pages 745-752
[75] M. Mambo, K. Usuda, and E. Okamoto, Proxy signatures: Delegation of the power to sign messages, IEICE Trans. Fundamentals, Sep. 1996, Vol. E79-A, No. 9, pp. 1338-1353
[76] S. Kim, S. Park, and D. Won. Proxy signatures, revisited. In: Information and Communications Security (ICICS'97), LNCS 1334, pp. 223-232, 1997. Berlin: Springer-Verlag, 1997.
[77] J. Zhang, J.Mao, "Another Efficient Proxy Signature Scheme in the Standard Model," Journal of Information Science Engineering, Vol. 27(4), 2011, pp.1249-1264

[78] M. Sliti, M. Hamdi, and N. Boudriga, “An Elliptic Threshold Signature Framework for k-Security in Wireless Sensor Networks”, The 15th IEEE International Conference on Electronics, Circuits, and Systems (ICECS 2008), August 31 – September 3, 2008, Malta.
[79] C. Zhe, L. Yuanning, “ Secure monitoring scheme for wireless sensor networks based on identity signcryption”, Computer Science and Automation Engineering (CSAE), Vol. 4, Pages. 309 – 312
[80] A. Sendonaris, E. Erkip, and B. Aazhang, "Increasing uplink capacity via user cooperation diversity", Proc. IEEE Int. Symp. Information Theory (ISIT), pp.156 1998
[81] A. Krohn. Et al,” Increasing connectivity in wireless sensor network using cooperative transmission. In 3rd International Conference on Networked Sensing Systems (INSS), Chicago, USA, May 31- June 2 2006.
[82] A. Sultan, et al. Network connectivity in Wireless Sensor Networks: a Survey, in Proc. PGNet, 2009
[83] Z. Zhu Han and Y. Sun, "Wave Cooperative Transmission Protocol for Underwater Acoustic Communications", IEEE International Conference on Communications, China, 2008
[84] M. J. Neely and R. Urgaonkar, "Optimal backpressure routing for wireless networks with multi-receiver diversity," Proc. Conf. Info. Sci. Sys., pp. 18-25, Jan. 2006.
[85] I. Maric and R. D. Yates, "Cooperative multihop broadcast for wireless networks," IEEE Journal of Selected Areas Communication, vol. 22, no. 6, pp. 1080-1088, August 2004.
[86] P. Herhold, E. Zimmermann, and G. Fettweis, “Cooperative multihop transmission in wireless networks,” Computer Networks Journal, vol. 49, October 2005.
[87] J. N. Laneman, D. N. C. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior", IEEE Trans. Inform. Theory, 2004.
[88] Y.-W. Hong , W.-J. Huang , F.-H. Chiu and C.-C. J. Kuo "Cooperative communications resource constrained wireless networks", IEEE Signal Process. Mag., vol. 24, pp.47 2007
[89] M. Yuksel and E. Erkip, “Diversity in relaying protocols with amplify and forward,” in Proc. IEEE GLOBECOM, Dec. 2003, vol. 4, pp. 2025–2029
[90] G. Kramer , M. Gastpar and P. Gupta "Cooperative strategies and capacity theorem for relay networks", IEEE Trans. Inf. Theory, vol. 51, pp.3037 2005
[91] M. Janani, "Coded Cooperation in Wireless Communications: Space-Time Transmission and Iterative Decoding", IEEE Trans. Sig. Proc., vol. 52, no. 2, pp.362 -371 2004
[92] T. E. Hunter and A. Nosratinia, "Diversity through Coded Cooperation", IEEE Trans. Wireless Commun., 2004
[93] A. Bletsas, H. Shin, and M. Win, “Outage-optimal cooperative communications with regenerative relays,” in Proc. Conf. Information Science Systems (CISS), 2006, pp. 632-637.
[94] A. Nosratinia and T. E. Hunter "Grouping and partner selection in cooperative wireless networks", IEEE J. Sel. Areas Commun., vol. 25, pp.369 2007
[95] A. Bletsas , A. Khisti , D. P. Reed and A. Lippman "A simple cooperative diversity method based on network path selection", IEEE J. Sel. Areas Commun., vol. 24, pp.659 2006
[96] J. Adeane, M. R. D. Rodrigues, and I. J. Wassell, “Optimum power allocation in cooperative networks,” in Proc. Postgraduate Research Conf. Electronics, Photonics, Commun. Networks, Computing Sci., Mar.- Apr. 2005, pp. 23–24.
[97] J. Yang, D. Gunduz, D.R. Brown III, and E. Erkip, "Resource allocation for cooperative relaying," in Proceedings of the Conference of Information Sciences and Systems (CISS 2008), (Princeton, NJ), Mar 2008.
[98] D. Gunduz and E. Erkip, “Opportunistic cooperation by dynamic resource allocation," IEEE Trans. Wireless Commun., vol. 6, pp. 1446-1454, Apr. 2007.
[99] M. Mahmoud and S. Shen, “A Secure and Efficient Incentive Routing Protocol for Multi-hop Wireless Networks”, CG 2010.
[100] L. Buttyan, J. Hubaux,” Enforcing service availability in mobile ad-hoc wans”, In Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC), Boston, MA, USA, August 2000.
[101] S. Zhong, Y.R. Yang and J. Chen, “Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad Hoc Networks,” Proc. IEEE INFOCOM "03 Conf., Mar.-Apr. 2003.
[102] A. Weyland and T. Braun, “CASHnet - Cooperation and Accounting Strategy for Hybrid Networks”, In Proceedings of 2nd Workshop on Modeling and Optimization in Mobile, Ad Hoc andWireless Networks (WiOpt), pp. 423–424. Cambridge University Press, Cambridge, UK, Mar. 2004.
[103] Y. Zhang, W. Lou, W. Liu, and Y. Fang, “A secure incentive protocol for mobile ad hoc networks,” Wirel. Netw., vol. 13, no. 5, pp. 569–582, Oct. 2007.
[104] J. Pan, L. Cai, X. Shen, and J. Mark, “Identity-Based Secure Collaboration in Wireless Ad Hoc Networks”, Computer Networks (Elsevier), Vol. 51, No. 3, pp. 853-865, 2007.
[105] M. Mahmoud and X. Shen, “DSC: Cooperation incentive mechanism for multi-hop cellular networks,” in Proc. IEEE ICC, Dresden, Germany, Jun. 14–18, 2009, pp. 569–574.
[106] M. Jakobsson, J. Hubaux, and L. Buttyan, “A micro-payment scheme encouraging collaboration in multi-hop cellular networks,” Proceedings of Financial Crypto 2003, Gosier, Guadeloupe, Jan. 2003.
[107] M. Jakobsson, and L. Yang, “Quantifying Security in Hybrid Cellular Networks”, ACNS Springer-Verlag Berlin/Heidelberg, Vol. 3531, pp. 350–363 , May 2005.
[108] L. Anderegg and S. Eidenbenz, “Ad hoc-VCG: A Truthful and Cost-efficient Routing Protocol for Mobile Ad Hoc Networks with Selfish Agents,” Proc. 9th Annual Int’l. Conf. Mobile Comp. and Net. (MobiCom 2003), Sept. 2003, pp. 245–59.
[109] M. Baker et al. “Participation Incentives for Ad Hoc Networks”, Project Report, University of California. Available: http://www.stanford.edu/yl314/ape/paper.ps, 2002
[110] J. Crowcroft et al, Modeling Incentives for Collaboration in Mobile Ad Hoc Networks. In Modeling and Optimization in Mobile Ad Hoc and Wireless Networks (2003).
[111] W. Wang, X. Y. Li, “Low-cost routing in selfish and rational wireless ad hoc networks”, IEEE Trans. Mobile Computing, vol. 5, May 2006, pp. 596 – 607
[112] S. Zhong and F. Wu, “A collusion-resistant routing scheme for noncooperative wireless ad hoc networks,” IEEE/ACM Transactions on Networking
[113] S. Zhong, L. Li, Y. G. Liu, and Y. R. Yang, “On designing incentive compatible routing and forwarding protocols in wireless ad-hoc networks,” in ACM MobiCom ’05, Aug. 2005.
[114] E. Ayanoglu et al., “Diversity Coding for Transparent Self-Healing and Fault-Tolerant Communication Networks,” IEEE Trans. Comm., vol. 41, no. 11, 1993, pp. 1677-1686.
[115] L. Zhou and Z. J. Haas. Securing ad hoc networks. IEEE Network Magazine, 13(6):24–30, November/December 1999
[116] Rajavelu Srinivasan, V. Vaidehi, K. N. Srivathsan, L. Ramesh Babu, C. Karunagaran: SeReRoM: Secured Reliable Routing Scheme for Multicasting. I. J. Network Security 5(1): 82-88 (2007)

[117] D. Somasundaram and R. Marimuthu, "A Multipath Reliable Routing for detection and isolation of malicious nodes in MANET," in In proceedings of International Conference on Computing, Communication and Networking, 2008.
[118] A. M. Abbas and B. N. Jain, Path Diminution in Node-Disjoint Multipath Routing for Mobile Ad Hoc Networks is Unavoidable with Single Route Discovery, International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC), Vol.5, No.1, 2010, pp.7-21.
[119] M. Bheemalingaiah., "Energy Aware Node Disjoint Multipath Routing in Mobile Ad Hoc Network", Journal of Theoretical and Applied Information Technology, JATIT, pp. 416-417, 2005-2009
[120] S. Upadhayaya and C. Gandhi. Node Disjoint Multipath Routing Considering Link and Node Stability protocol: A characteristic Evaluation. IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 1, No. 2, January 2010,18-25
[121] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults', Proc. 26th FOCS, 1985, pp. 383-395
[122] M. Tompa and H. Woll, "How to share a secret with cheaters," in Lecture Notes in Computer Science 263; Advances in Cryptology: Proc. Crypto '86, A. M. Odlyzko, Ed., Santa Barbara, CA, Aug. 11-15, 1986, pp. 133-138. Berlin: Springer- Verlag, 1987.
[123] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, Advances in Cryptology, CRYPTO '91, volume 576 of Lecture Notes in Computer Science, pages 129–140. Springer-Verlag, 1992.
[124] R. Ostrovsky and M. Yung, How to withstand mobile virus attacks, Proc. of the 10th ACM Symp. on the Princ. of Distr. Comp., 1991, pp. 51–61.
[125] A. Herzberg, S. Jarecki, H. Krawczyk, M. Yung, Proactive Secret Sharing, or: how to cope with perpetual leakage, Advances in Cryptology -Crypto 95 Proceedings, Lecture Notes in Computer Science Vol. 963, D. Coppersmith ed., Springer-Verlag, 1995, pp. 339-352.
[126] D. Stinson and R. Wei, Unconditionally Secure Proactive Secret Sharing Scheme with Combinatorial Structures, SAC'99. Lecture Notes in Computer Science, vol. 1758, pp. 200–214, 1999.
[127] Tartary C., Wang H.: Dynamic threshold and cheater resistance for shamir secret sharing scheme. In: Proceedings of Inscrypt'06, LNCS, vol. 4318, pp. 103–117. Springer-Verlag (2006).
[128] K. Martin, J. Pieprzyk, R. Safavi-Naini, H. Wang Changing thresholds in the absence of secure channels Australian Comput. J., 31 (1999), pp. 34–43
[129] Y. Desmedt and S. Jajodia. Redistributing secret shares to new access structures and its applications. Tech Report TR-97-01, GMU, 1997.
[130] K. M. Martin, R. Safavi-Naini, and H. Wang, "Bounds and techniques for efficient redistribution of secret shares to new access structures," Comput. J., vol. 42, no. 8, pp. 638–649, 1999.
[131] R. Steinfeld, J. Pieprzyk and H. Wang, "Lattice-based threshold changeability for standard shamir secret-sharing schemes," Adv. in Cryptology - ASIACRYPT 2004, Lecture Notes in Comput. Sci., 3329, pp.170–186, 2004.
[132] Z.-W. Tan, Z.-J. Liu and M.-S. Wang: On the security of some nonrepudiable threshold proxy signature schemes. ISPEC 2005, Lecture Notes in Computer Science 3439, pp.374-385, Springer-Verlag, 2005.
[133] K. Kim and D. Nyang Security Analysis of a Threshold Proxy Signature Scheme IACR Cryptology ePrint Archive 2010: 400 (2010)
[134] K. Zhang, "Threshold proxy signature schemes", Proceedings of 1997 Information Security Workshop, 1997, Japan, pp. 191-197
[135] H.M. Sun. An efficient nonrepudiable threshold proxy signature scheme with known signers. Computer Communications, 22(8), 717–722. (1999).
[136] C. L. Hsu, T. S. Wu, and T. C. Wu, Improvement of threshold proxy signature scheme, Appl. Math. Compu.,vol. 136, pp. 315-321, 2003.
[137] K. Shum, V.K Wei, "A strong proxy signature scheme with proxy signer privacy protection". In: 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2002), pp. 55–56. IEEE, Los Alamitos (2002).
[138] N. Lee and M. Lee, "The Security of a Strong Proxy Signature Scheme with Proxy Signer Privacy Protection", Applied mathematics and computation, 16 (1), 2005, pp. 807–812.
[139] F. Zhang, R. Safavi-Naini, C.-Y. Lin, New proxy signature, proxy blind signature and proxy ring signature schemes from bilinear pairing. Cryptology ePrint Archive, Report 2003/104 (2003)
[140] J. Hu, J. Zhang, "Cryptanalysis and improvement of a threshold proxy signature scheme", Computer Standards & Interfaces (2008)
[141] Xuan Hong, Kefei Chen Secure multiple-times proxy signature scheme Computer Standards & Interfaces (2007)
[142] Yong Yu, Chunxiang Xu, Xinyi Huang, Mu Yi An efficient anonymous proxy signature scheme with provable security Computer Standards and Interfaces, 31 (2) (2009), pp. 348–353
[143] S. Dehnie, H. Sencar, and N. Memon, "Cooperative diversity in the presence of misbehaving relay: Performance analysis," in IEEE Sarnoff Symposium, 2007.
[144] A. Aksu, P. Krishnamurthy, D. Tipper, and O. Ercetin. On security and reliability using cooperative transmissions in sensor networks. In Proceedings of the 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing, pages 1{10, 2010.
[145] L. Zhang, L. Cong, K. Zhao, Yang, and H. Zhang, "Competitive resource sharing based on game theory in cooperative relay networks," ETRI Journal, vol. 31, pp. 89–91, 2009
[146] L. Chen, L. Libman, J. Leneutre, "Conflicts and Incentives in Wireless Cooperative Relaying: A Distributed Market Pricing Framework," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 5, pp. 758-772, May 2011
[147] G. Zhang, L. Cong, K. Yang, et al, "Competitive Resource Sharing Based on Game Theory in Cooperative Relay Networks," ETRI Journal, 2009, 31(1), pp. 89-91.
[148] B. Wang, Z. Han, and K. Liu. Distributed relay selection and power control for multiuser cooperative communication networks using buyer/seller game. In IEEE INFOCOM Proceedings, pages 544–552, 2007.
[149] D. Yang, X. Fang, and G. Xue, "Truthful Auction for Cooperative Communications," Proc. ACM MOBIHOC'11, pp. 89–98.
[150] S. Marti, T.J. Giuli, K. Lai, M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking, 2000, pp. 255–265
[151] S. Buchegger and J.-Y. L. Boudec, "Performance analysis of the CONFIDANT protocol: Cooperation of nodes - fairness in dynamic ad-hoc networks," in Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC). IEEE, June 2002.
[152] S. Buchegger and J.-Y. Le Boudec. A Robust Reputation System for Peer-to-Peer and Mobile Ad-hoc Networks. Proceedings of P2PEcon 2004, Harvard University, Cambridge MA, U.S.A., June 2004.
[153] P. Michiardi , R. Molva, Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks, Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security, p.107-121, September 26-27, 2002
[154] N. Jiang, K. Hua, and D. Liu, "A Scalable and Robust Approach to Collaboration Enforcement in Mobile Ad-Hoc Networks", Communication and Networks, Vol. 9, Part 1, pp. 56-66, 2007.

[155] M. Lee, X. Ye, S. Johnson, D. Marconett, C. Vsk, R. Vemuri, and Yoo, S. J. B, “Cognitive security management with reputation based cooperation schemes in heterogeneous networks”. In IEEE symposium on computational intelligence in cyber security (CICS). Nashville, TN, March 2009.
[156] S. Ganeriwal and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks", *Proc. ACM Security for Ad-Hoc and Sensor Networks*, pp.66 -67 2004.
[157] H. Chen, H. Wu, X. Zhou, C. Gao, Agent-based trust model in Wireless Sensor Networks, Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD 03, 2007, pp. 119–124
[158] Z. Han and Y. Sun, “Securing Cooperative Transmission in Wireless Communications”, in Procs. of First Workshop on the Security and Privacy of Emerging Ubiquitous Communication Systems, Pennsylvania, USA, Aug. 2007.
[159] R. Changiz, H. Halabian, , F. Yu, R., I. Lambadaris, H. Tang, and Mason, P. C. (2010, November), “Trust establishment in cooperative wireless networks.” In Proceedings of IEEE Milcom’10.
[160] Y. Mao, M. Wu, "Tracing Malicious Relays in Cooperative Wireless Communications," IEEE Transaction on Information Forensics and Security, vol. 2, no. 2, pp. 198-212, June 2007.
[161] S. Dehnie, H. Senear, and N. Memon, "Detecting malicious behaviour in cooperative diversity," in Proc. Conference on Information Sciences and Systems, Baltimore, USA, Mar. 2007.
[162] Li-Chung Lo, Wan-Jen Huang, “Misbehaviour Detection without Channel Information in Cooperative Networks”, VTC Fall 2011: 1-5.
[163] Q. Mahmoud, "Cognitive Networks: Towards Self-Aware Networks", John Wiley and Sons, 2007.
[164] OMNet++, http://www.omnetpp.org/, June 2012.
[165] NS3, http://www.nsnam.org/, June 2012.
[166] Y. Sun "A Trust Evaluation Framework in Distributed Networks: Vulnerability Analysis and Defense Against Attacks", Proc. IEEE INFOCOM 2006, 2006.