Evaluation of a game-theoretical protocol for certificate revocation in vehicular networks

Thesis submitted in partial fulfillment of the requirements for the degrees of
Master of Science in Telecommunication Engineering
and
Master of Science in Computer Science

by

Ignacio Llatser Martí

Supervised by
Dr. Mohammad Hossein Manshaei
and
Maxim Raya

2008

Laboratory for computer Communications and Applications 1 (LCA1)
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne

Abstract

Vehicular networks are an emerging instantiation of ad-hoc networks that will allow cars to exchange information wirelessly in order to improve road safety. Providing security to these networks remains a crucial challenge: undetected attackers may trigger an accident by sending messages with bogus information. In order to prevent malicious nodes from abusing the system, a distributed protocol for certificate revocation is needed.

In this thesis, we evaluate RevoGame, a promising recently-published protocol based on game theory. With this aim in view, we perform simulations using ns-2 in a realistic scenario, comparing RevoGame with the already-existing mechanisms LEAVE and Stinger. We also investigate the effect of several parameters, such as the probabilities of detection and of false positives, on the behavior of the protocol.

Based on our results, we then redesign certain aspects of RevoGame, and show that the optimized protocol globally outperforms LEAVE and Stinger. Nonetheless, there is still room for improvement; we finally outline Dynamic Voting, a new protocol based on voting with a variable number of votes that yields better results in the chosen scenario.

Acknowledgements

First and foremost I would like to express my deep and sincere gratitude to Dr. Mohammad Hossein Manshaei and Maxim Raya, who jointly supervised this master thesis. Their invaluable insights and guidance have been a source of inspiration throughout the development of this project. Many of the ideas contained in this work originated in the stimulating discussions that we held together.

I am also very grateful to Prof. Jean-Pierre Hubaux for giving me the opportunity to do the master thesis in the Laboratory of computer Communications and Applications 1 (LCA1) at EPFL. He gave me expert advice at crucial stages of the project. I greatly appreciate as well the financial support offered by LCA1 for the development of this thesis.

I would also like to take this opportunity to acknowledge all the friends that have helped me since I began studying at the Technical University of Catalonia (UPC), and so have indirectly contributed to this thesis. Special thanks go to Josep Miquel Jornet, Marc Oriol, Anna Papió, Òscar Senén and Franchesca Salcedo for their valuable friendship.

Last but certainly not least, a big thank you goes to my parents, my brother Franc and the rest of my family for their unconditional support throughout my life.

Contents

List of Figures

Chapter 1

Introduction

1.1 Project overview

In this project, we evaluate the performance of RevoGame, a revocation procotol based on game theory designed specifically for ephemeral networks. With this purpose, we implement highly realistic simulations; results show that RevoGame performs better than other previously-proposed protocols. In the simulations, we also investigate the effect of different parameters on the behavior of RevoGame.

Based on simulations' results, we have decided to redesign some aspects of RevoGame. We show how these modifications improve the performance of the protocol. After finding the weaknesses of RevoGame, we outline Dynamic Voting, a new protocol that might be more suitable for vehicular networks.

This thesis is organized as follows. In chapter 1, we introduce the concept of vehicular networks, explain why certificate revocation is important in these networks and show the adversary model that will be used. Chapter 2 presents the certificate revocation protocols that will be later analyzed and compared. In chapter 3, we describe some of the design choices taken during the development of the RevoGame protocol, and define a new protocol called Dynamic Voting. The results of the simulations performed are shown and explained in chapter 4. Finally, in chapter 5 we conclude the report and outline further work.

1.2 Vehicular Ad-Hoc Networks

Vehicular Ad-Hoc Networks (VANETs) are an emerging form of Mobile Ad-Hoc Networks (MANETs). The objective of vehicular networking is to improve the safety of transportation systems by enabling communication among vehicles and between vehicles and roadside equipment. The information transmitted consists mainly of warnings about accidents and/or traffic conditions

(e.g. congestion). Non-safety applications may be included as well, such as high-speed tolling or mobile infotainment.

Vehicular networks are expected to implement different wireless technologies. One of the most promising is Dedicated Short Range Communications (DSRC) [3], a short to medium range wireless protocol specifically designed for automotive use. Other technologies such as cellular systems may also be included in VANETs.

The creation of high-performance, secure, reliable and scalable VANETs presents enormous research challenges. Many of the existing protocols for regular ad-hoc networks have to be rethought due to the particularities of vehicular networks in terms of mobility, requirements and scenarios.

Moreover, VANETs are currently a very active field of research. Throughout the world, many projects are being carried in government, industry and academia devoted to VANETs. Consortia such as Vehicle Safety Consortium (US), Car-2-Car Communication Consortium (Europe) [1], and Advanced Safety Vehicle Program (Japan) are actively working on the development of vehicular networks. Standardization efforts such as the IEEE 802.11p standard, also known as Wireless Access in the Vehicular Environment (WAVE) [10], show the current interest in VANETs. Vehicular networks are seen as a promising technology which is set to revolutionize the concept of transportation as we know it today. Figure 1.1 [1] shows the communication among vehicles in a Vehicular Ad-Hoc Network.

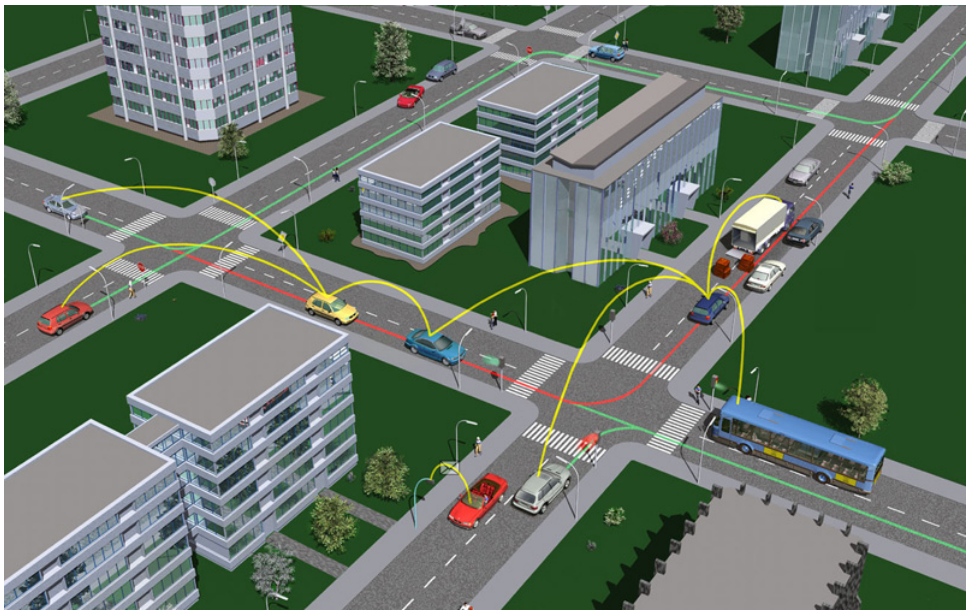


Figure 1.1: Vehicular Ad-Hoc Network

1.3 Certificate revocation

A crucial issue in the context of wireless ad-hoc networks is the ability to revoke malicious or malfunctioning devices. In this context, Public Key Infrastructure (PKI) is widely recognized as one of the most effective techniques to provide security in wireless networks [11]. The traditional way to remove malicious nodes from the network is by means of Certificate Revocation Lists (CRLs). This approach suffers from bad performance, poor scalability and the need for online connectivity to the Certification Authority (CA). In vehicular networks, there is no guarantee that every device will be connected to the CA at all times, especially in the early stages of deployment of the network; hence, CRLs may not be suitable for certificate revocation in VANETs.

As a consequence, there is a need for alternative solutions to provide a security architecture to VANETs. In this sense, several protocols revolving around the concept of *local revocation*, which allows the temporary eviction of misbehaving neighbors without the presence of a CA, have been proposed. Based on the local revocation, the CA may eventually decide to permanently revoke the malfunctioning nodes from the network.

In the next chapter, we will provide a brief overview of three of these protocols. The first one is LEAVE [7], based on the idea of removing attackers by means of voting. The next one is Stinger [5], which introduces the idea that excluding a node from the network has a certain cost. Elaborating on this idea, the RevoGame protocol [6] appears, a game-theoretic approach which aims to take the best of the two previous techniques while minimizing their weaknesses.

1.4 Adversary model

In our work, as in [5], we consider two different attacker strategies. First, adversaries may disseminate *false information* in the network. Their motivation may be either malicious (e.g., sending fake braking information to trigger an accident) or selfish (e.g., claiming an accident to clear the road).

The second attack is the *exclusion mechanism abuse*. Indeed, one of the main issues when designing a local revocation scheme is the possibility of abuse of the protocol by malicious devices. That is, we assume that attackers will try to exploit the system by removing as many good nodes as possible before being revoked themselves.

Chapter 2

Certificate revocation in vehicular networks

This chapter presents the three protocols for certificate revocation that will be later evaluated and compared.

2.1 LEAVE

LEAVE (Local Eviction of Attackers by Voting Evaluators) is based on the idea of voting against malicious nodes. Vehicles detecting an errant device add it to its corresponding accusation database, and broadcast a *warning* message to alert all vehicles in range of the attacker. Any vehicle receiving the message adds the accused node to its own accusation database as well.

Once a device has collected a threshold number of votes against a given node, it adds the accused node to a local blacklist. This means that whenever a node receives a message from an ignored device, the message will be rejected and a *disregard* message will be broadcast to instruct all neighboring nodes to ignore the attacker's messages. This message will contain the *supporting signatures* of all the devices which have voted against the misbehaving node. Hence, nodes can be made aware of the presence of attackers before interacting with them.

One of the main disadvantages of the LEAVE protocol is its weakness against organized groups of malicious devices. Indeed, a sufficiently large group of attackers, by voting coordinately against good nodes, may eject any of them from the network at will.

2.2 Stinger

Stinger introduces the concept of suicide attacks in VANETs. Whenever a vehicle detects a malicious device, it broadcasts a *sting* message against it. A

sting is equivalent to a self-sacrifice: all vehicles receiving it will ignore both the accusing and the accused devices. By imposing such a high cost on the revocations, the possibility of the attackers to abuse the system is considerably restricted.

One of the drawbacks of Stinger is that, in networks with high mobility such as VANETs, it is possible that many good nodes have to self-sacrifice in order to exclude a single malicious node. However, it is important to note that no single device will ignore two honest nodes because of the same attacker. This is achieved because every device maintains a local blacklist of revoked attackers, and only the first sting sender is ignored for each accused device.

2.3 RevoGame

The RevoGame protocol introduces a completely new approach to the issue of certificate revocation. It applies concepts of game theory to define a model of revocation in ephemeral networks. The main characteristics of ephemeral networks are their large scale and the high mobility of the wireless devices. While the scope of the protocol is not limited to vehicular networks, we will consider its application to VANETs.

In this model, whenever a non-ignored attacker is detected by a group of nodes, a *dynamic (sequential) game* [4] is started. Every node which detected the attacker will be a player in the game. The game will have as many stages as the number of participants. At every stage of the game, every player will take one of three actions. First, a player can *abstain* from the local revocation procedure. This means that the player is not willing to contribute in the eviction of the attacker, and expects the other nodes to revoke it. The second strategy is to send a *vote* against the malicious device. The calculation of the required number of votes to evict an attacker is detailed in section 3.4. Finally, a player can *self-sacrifice* in order to revoke unilaterally the attacker.

In terms of costs, we may consider two different types of revocation games. On the one hand, we have *games with fixed costs*, where the attacker cost is fixed. In this model, players tend to leave the revocation decisions to the last stages of the game. This is a very risky behaviour in ephemeral networks, as some of the players will likely move out of range before the end of the game, and their decision will not reach all the other players. One the other hand, we have *games with variable costs*, which assume that the cost inflicted by an attacker increases with time. These games encourage players to begin the revocation process (by voting or self-sacrifice) in the early stages of the game. In this work, we will only consider games with variable costs.

Chapter 3

Design issues

In this chapter, we describe some of the choices taken during the design of the RevoGame protocol. We also outline the changes made with regard to the original protocol, their motivation and how they improved the performance of the revocation scheme.

3.1 Estimation of the number of neighbors

An important parameter of RevoGame is the number of neighbors of a given device, which we denote by N. The estimation of this parameter is done locally by every node. For this purpose, every node has a *neighbor database* where it stores information about all the devices previously heard.

Every time a node receives a regular broadcast from another device, it stores the position of the sender, the timestamp of the message and whether the sender is an attacker or not. It is important to note that malicious nodes consider benign nodes as attackers, and vice versa. With all this information, the task of calculating the estimated number of neighbors is reduced to counting the number of benign nodes in the neighbor database whose position is located in the intersection of the attacker's transmission range and the node's own. For this calculation, only devices heard in the last 300 milliseconds are considered. This temporal restriction prevents nodes from counting the devices that were once the node's neighbors but have already moved out of range.

3.2 Estimation of the number of players

An issue that we identified thanks to simulations was that a high number of games did not finish (and thus the attacker was not revoked). This was due to a bad estimation of the number of players in the game. This value was originally estimated locally as $p_d N$, i.e., the probability of detecting an attacker multiplied by the number of good nodes in the neighborhood.

However, this calculation did not take into account the fact that there is a high chance that some of the node's neighbors will be in an *inactive* state. This may be caused by two reasons: a) the node has already ignored the attacker, and b) the node has participated in a game in the last 300 ms (see section 3.6). Therefore, there will be a group of nodes that will not join the game, and the number of players will thus be over-estimated. As a consequence, many games will not finish, because the nodes will keep waiting forever for the actions of the players which in reality are not participating in the game.

So, we identified the inaccurate estimation of the number of players in a game as the cause of unfinished games. A precise local estimation of this value is hard to obtain, as a node ignores a priori how many of its neighbors are inactive (i.e., they are in one of the two situations previously described).

In order to solve this issue, we have adopted a strategy that concerns the estimation of the number of active players. As we want to minimize the number of packets sent to the network, we avoided solutions consisting of periodic broadcasts of every node's state; instead, we aim to perform a better local estimation. In order to take into account the chance of a node playing a game, we define the probability of being active p_a as the fraction of times a node starts a game after detecting an attacker, that is:

$$p_a = \frac{\text{number of games played}}{\text{number of times an attacker has been detected}}$$

Contrary to our initial thoughts, we observed that p_a usually has a very low value; this means that most of the times nodes are inactive for one of the reasons described above. The probability of being active will have, as we will see in section 3.4, a crucial influence on the behavior of the protocol.

Taking the probability of being active into account, we approximate the number of players as $p_d p_a N$, which leads to a dramatically improved estimation of this value. We show the improvement in the performance of RevoGame brought by this modification in section 4.5.4.

Besides this strategy, we initially set an upper limit on the duration of a game. The idea behind this was to allow nodes to start a new game if the current one has not finished after a time limit is reached. However, after introducing a minimum time between games (see section 3.6), the upper limit on the duration of a game proved to be unnecessary. Indeed, even if an unfinished game was quickly aborted, its players still had to wait for the minimum interval between games to expire in order to start a new game. Consequently, we eventually decided to remove the upper limit on the duration of games from the protocol.

3.3 Estimation of the number of attackers

The estimation of M, the number of malicious devices, is done in a similar way to the estimation of the number of neighbors (see section 3.1). There is, however, a crucial difference between them. When calculating the number of attackers, we take into account all the misbehaving devices heard, not just the ones in our neighborhood. We hence consider a worst-case scenario, in terms of the attackers' capabilities to harm the benign nodes.

As it happens with the calculation of the number of neighbors, attackers calculate M as the number of benign devices heard.

3.4 Estimation of the optimal number of votes

One of the key parameters of RevoGame is the required number of votes to revoke an attacker in case that the chosen strategy is voting. The protocol introduces the notion of *social cost* to cope with this issue. The social cost represents the damage caused by the attackers to all the benign nodes in the system. It was originally expressed by the following formula:

$$C_{orig} = nv + p_d p_a N \frac{M-1}{n} + p_{fp}^n (1 + nv) + N (n-1) \delta \tag{3.1}$$

In this equation, v is the cost of voting and δ is the cost inflicted by an attacker in a single stage of the game. The probability of detecting a false positive is represented by p_{fp}. The optimal number of votes n is obtained by minimizing the social cost.

After performing some preliminary simulations, we observed that the obtained number of votes was in many cases greater than the number of players in the game. This forced many benign nodes to self-sacrifice in order to revoke the attacker, a behavior that we want to minimize. So, we derived a new, simplified cost expression from which to obtain the optimal number of votes.

The idea behind the new social cost is that, on the one hand, we want to minimize the required number of votes to revoke an attacker, in order to evict it faster from the network. On the other hand, we seek to reduce the attackers' ability to abuse the system, by minimizing the number of benign nodes they would be able to revoke in the worst case. So, we define the cost as the addition of these two factors:

$$C = n + \frac{M}{n} \tag{3.2}$$

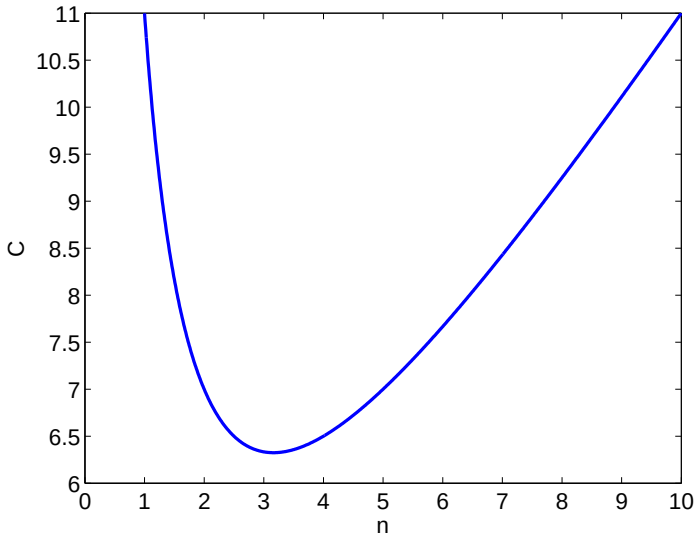


Figure 3.1: Social cost as a function of the number of votes

Figure 3.1 shows C as a function of the number of votes n, in a scenario with 10 attackers. As we can see, in order to minimize C, we must find the stationary point using Fermat's theorem; i.e., taking the first derivative of C with respect to n and setting it to zero:

$$\frac{\partial C}{\partial n} = 1 - \frac{M}{n^2} = 0 \tag{3.3}$$

This gives us a number of votes $n = \sqrt{M}$. This value, albeit lower than the one obtained with the original social cost, still exceeds in some cases the number of players, and hence it requires nodes to self-sacrifice. In order to avoid this undesired behavior, we set the optimal number of votes to the minimum between the previous value and the number of players in the game, that is:

$$n = \min\left(p_d p_a N, \sqrt{M}\right) \tag{3.4}$$

With this number of votes, we virtually eliminate the self-sacrifices from the protocol, as in every game there will be enough players to revoke an attacker by voting. Another consequence of this change, as we have observed thanks to simulations, is that the optimal number of votes is 1 in most of the cases. The reason is that the probability of a node being active p_a (defined in section 3.2) is usually very low. In other words, most games have just one player (as every other node is inactive), and then the best decision is to revoke the attacker with one vote.

3.5 Sequential games

An important challenge encountered during the design of the protocol was how to implement sequential games in a wireless network. In sequential games, there is an explicit order in which the players participate in the game. Moreover, every player knows the actions of the preceding players, and its decision is influenced by this knowledge.

We recall that, in the RevoGame protocol, a game is started whenever a group of nodes detect an attacker. Then, every active node which observed the attacker has to broadcast its decision (abstain, vote or self-sacrifice) sequentially to the rest of the players. However, the detection of an attacker by the various players happens at the same time, so we need to introduce an order in the sequence of decisions.

In order to do this, we define a random backoff (between 0 and 20 ms) that every node has to wait before sending its move. This way, we minimize the probability that two or more nodes transmit their strategy simultaneously. If a node receives another device's decision before sending its own, it plays the game again (taking into account the newly received information) and broadcasts the new outcome.

In short, using a random backoff we introduce an order in the decisions of the various players in the game. Most importantly, the decision of every node will be crucially influenced by the previous nodes' moves.

3.6 Minimum time between games

After testing the first version of the protocol, we also observed that the number of revoked benign nodes was very high. By means of preliminary simulations, we quickly identified the problem that caused this issue: attackers could play a game as often as they wanted, and so they were able to easily revoke many good nodes.

In order to minimize this undesired effect, we have enforced a minimum time between games. That is, a node will not be able to play a game as often as desired; once it plays, it will have to wait for a given period of time before being able to start a new game. We have set this minimum time to the same value as the interval between regular broadcasts (300 ms). This way, we can assure that any node will send a regular broadcast between any two games. Thanks to this, benign nodes surrounding an attacker will be able to detect it before the malicious node is able to revoke more benign devices.

As expected, following this modification of the protocol, the number of revoked benign nodes was significantly reduced, as it is shown in section 4.5.4.

3.7 Dynamic Voting

As we have discussed in section 3.4, we have come to the conclusion that the key aspect in the design of the optimal number of votes is to avoid self-sacrifices, as they cause many benign nodes to lose their keys. In order to achieve this, we set an optimal number of votes which prevented nodes from choosing the self-sacrifice strategy.

Instead of fixing somewhat artificially the optimal number of votes so that it never exceeds the number of players, we can take a completely different approach: design a new protocol, which we call *Dynamic Voting*, based on voting with a variable number of votes. We eliminate the strategies of abstain and self-sacrifice, and leave voting as the only possible move. The optimal number of votes n is calculated as the value that minimizes the social cost derived in section 3.4, i.e. $n = \sqrt{M}$. This modified protocol, albeit simple, yields surprisingly good results, which we show in section 4.5.6.

Chapter 4

Performance evaluation

In order to evaluate the performance of RevoGame, we have simulated the protocol in a vehicular network. We explain in this chapter the parameters involved in the simulations and show the results obtained.

4.1 Simulation parameters

Two important values in our simulations are the probability of detection and the probability of false positives. Even though they have been introduced in the previous chapter, we will describe them here in more detail.

We denote the probability of detection by p_d. Whenever an attacker sends a regular broadcast, every benign node in its transmission range will detect the presence of the attacker with a probability p_d. Then, all the active benign devices that have spotted the attacker will start a game to revoke it.

On the other hand, whenever a benign node receives a regular broadcast from another good device, it will mistake the sender for an attacker with a probability p_{fp}, the probability of false positives. The receiver will then start a game to evict the sender from the network.

We assume, as a worst-case scenario, that attackers know the identities of all other malicious nodes in advance. We thus consider them to have perfect detection capabilities, i.e. $p_d = 1$ and $p_{fp} = 0$.

Lastly, it is important to note that we only take p_d and p_{fp} into account when receiving regular broadcasts. The detection capability of the devices is based on the analysis of the data contained in regular broadcasts, so we assume that nodes cannot determine, upon receiving a revocation message, whether the sender is misbehaving or not.

4.2 Costs

In game theory, the notion of costs influences the decision of every player in a game. As we said in section 2.3, we assume that the attacker cost increases with time (games with variable costs). All costs are represented in terms of keys.

RevoGame mainly considers two types of costs: the cost incurred by an attacker and the cost of participation in revoking the attacker. On the one hand, we have the cost of receiving a regular broadcast from a non-ignored malicious node, which is denoted by δ. On the other hand, there is the cost of each of the different players' strategies: abstain, vote and self-sacrifice.

It is straightforward to assume that abstaining from the game does not cost the players anything. Self-sacrifice has a cost of 1 key, as by doing so the player completely loses the ability to use the sacrificed key. The cost of voting is denoted by v. We make the reasonable assumption that $v < \delta < 1$.

As we saw in chapter 3.4, the behavior of the protocol is independent of the values of the different costs (with the aforementioned restrictions). Therefore, we have set v and δ to the values of 0.02 and 0.1 respectively for all simulations.

4.3 Scenario

One of the main objectives of this project was to evaluate the RevoGame protocol in conditions as realistic as possible. For this reason, we have chosen a city scenario, where 303 vehicles move at an average speed of 50 km/h. We have considered an area of 6.8 km $\times$ 5.5 km in a neighborhood of Manhattan. The mobility traces have been generated using the VANET simulation framework TraNS [9]. Figure 4.1, rendered in Google Earth using TraNS, shows the scenario used.

4.4 Configuration

We have performed all the simulations using the popular network simulator ns-2. It allowed us to implement the different certificate revocation protocols in C++, and configure the parameters of the simulations using the scripting language Tcl.

In our simulations, we use the Nakagami propagation model, which is described in [8]. We have chosen the MAC model 802.11Ext implemented in ns-2.33, adjusting its parameters to comply with the 802.11p specifications. Each vehicle sends regular broadcasts every 300 ms over a transmission range

Figure 4.1: Simulation scenario

of approximately 150 m, conforming to the DSRC specification (i.e. the communication range is equal to the distance the vehicle can cover in 10 seconds at its current speed).

The simulations have been performed on a cluster of 164 computers, managed by the Condor High Throughput Computing System [2]. All simulations have been averaged over 50 runs with 95% confidence intervals. The duration of each simulation is 145 seconds.

4.5 Simulation results

We describe in this section the different simulations that have been performed. We have set $p_d = 0.8$ and $p_{fp} = 10^{-4}$ for benign nodes in all simulations (unless otherwise noted). Even though the value of the probability of false positives may seem very low, it is important to remember that p_{fp} is applied to every message received from a benign node. Due to what is known as the *false positive paradox*, a higher p_{fp} would cause many good nodes to be revoked, even if the number of attackers was small. A way to achieve such a low p_{fp} might be, instead of accusing a suspicious node the first time that misbehavior is detected, to postpone the accusation until a certain number of bogus messages have been received from the suspicious node.

4.5.1 Comparison with LEAVE and Stinger

The first simulations that we performed involved contrasting RevoGame with the previously-proposed protocols for certificate revocation in vehicular

networks (which were introduced in chapter 2). The first one is LEAVE, a voting scheme with a fixed number of votes; we chose a value of $n = 5$. The second protocol that we compare with RevoGame is Stinger, an approach based on self-sacrifices.

Firstly, we plot the *percentage of revoked attackers* as a function of the number of malicious nodes in the scenario. We consider an attacker as revoked if it has been ignored by at least one benign node. Of course, the objective of any protocol for certificate revocation is to maximize the number of revoked attackers.

As we see in Figure 4.2, both Stinger and RevoGame revoke virtually all attackers, independently of their number. This is due to their capability to quickly evict malicious nodes (Stinger by self-sacrificing and RevoGame by sending revocation messages with one vote). LEAVE, however, performs badly when the number of attackers increases, as at least 5 nodes must detect an attacker in order to evict it from the system.

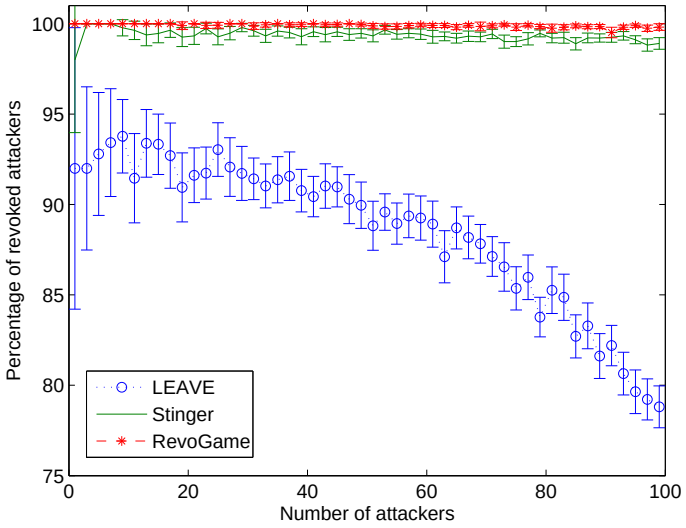


Figure 4.2: Percentage of revoked attackers as a function of the number of attackers in the scenario

The second result that we want to compare is the *percentage of revoked benign nodes*. As before, we consider a node as revoked if it has been ignored by at least another benign node. Good nodes may be evicted for two reasons: a) abuse of the system by attackers, and b) false positives. As we observe in Figure 4.3, LEAVE has the best performance in terms of revoked benign nodes. This is an expected result, as the requirement of 5 votes makes it difficult for attackers to abuse the system, and it is also very unlikely that a node is revoked because of false positives.

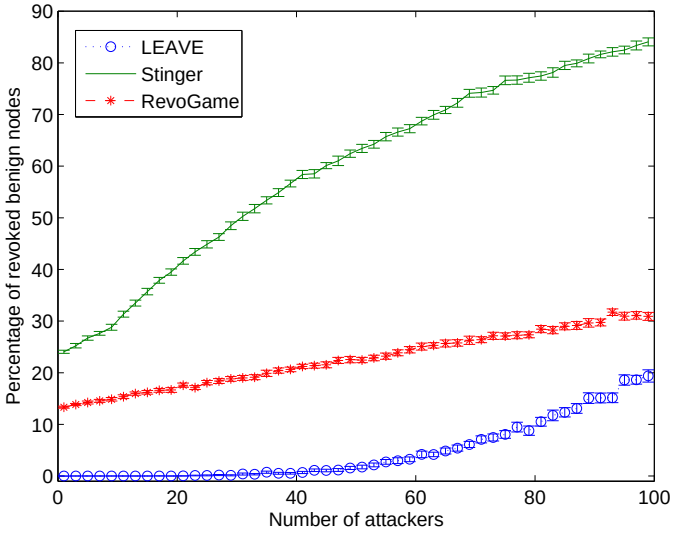


Figure 4.3: Percentage of revoked benign nodes as a function of the number of attackers in the scenario

Clearly Stinger has the highest number of revoked good nodes, as devices need to self-sacrifice in order to evict an attacker. RevoGame, as explained in section 3.4, does not use self-sacrifices, and hence it performs better. It is also worth to note that, although RevoGame revokes more benign nodes than LEAVE, it scales better when the number of attackers increases. This is due to its superior capabilities of quickly removing attackers, which greatly mitigate the ability of attackers to abuse the system.

We will explore next the *social cost* of each protocol. We compute it using the following formula:

$$C = N_m\delta + N_v v + N_s \tag{4.1}$$

where N_m is the number of regular broadcasts sent by attackers and received by benign nodes during the simulation, N_v is the total number of votes sent by benign nodes, and N_s is the number of good nodes that self-sacrifice. We thus notice that the social cost strongly depends on the chosen values for δ and v.

The social cost of Stinger is the highest due to the high cost of self-sacrifice, as we see in Figure 4.4. The cost of RevoGame is greater than that of LEAVE when the number of attackers is small, but the opposite happens with a higher number of attackers. This behavior confirms the fact that RevoGame is more resilient in scenarios with many malicious nodes.

The next results concern the *maximum time to revocation*. We have defined it as the moment when the last revoked attacker is evicted from the network.

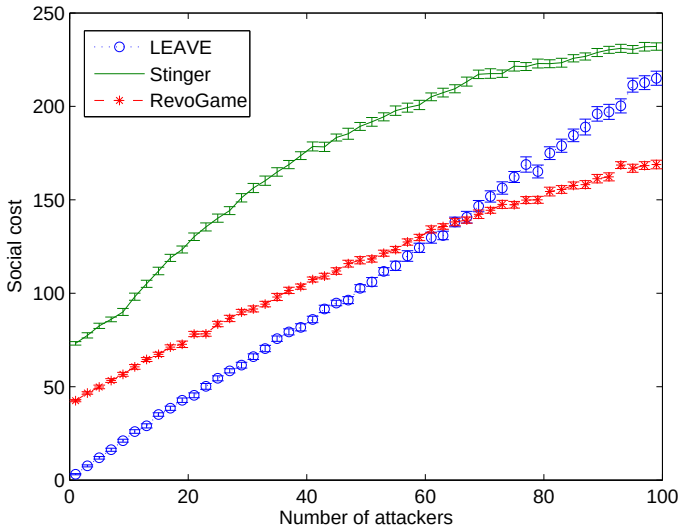


Figure 4.4: Social cost as a function of the number of attackers in the scenario

Intuitively, it reflects how quickly every protocol evicts the misbehaving nodes. The graph in Figure 4.5 confirms our intuition: LEAVE is the slowest protocol, as 5 votes are needed to evict an attacker. On the other hand, Stinger is not as fast as RevoGame, because the shortage of non-revoked benign nodes (due to self-sacrifices) lengthens the revocation of malicious nodes.

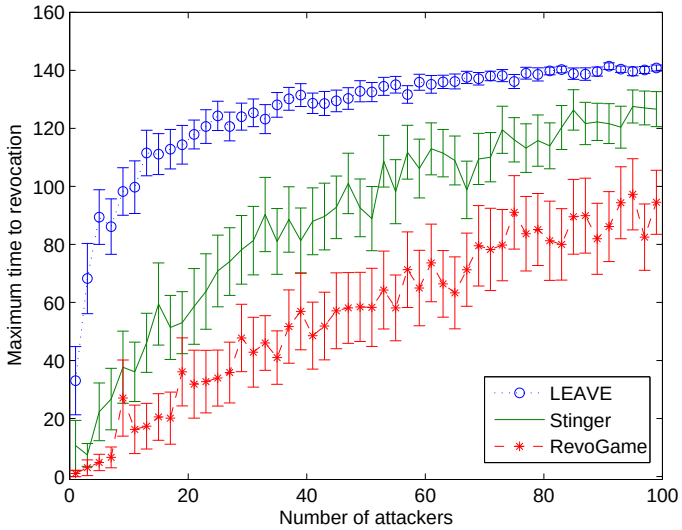


Figure 4.5: Maximum time to revocation as a function of the number of attackers in the scenario

When we counted the numbers of revoked attackers and benign nodes, we did not take into account how many benign nodes are actually ignoring each

of these nodes. We consider this in Figure 4.6, where we plot the *average number of benign nodes ignoring every attacker*. We see again that RevoGame outperforms the other two strategies, which behave similarly. On average, using RevoGame every attacker is ignored by 3 more benign nodes compared to employing LEAVE or Stinger.

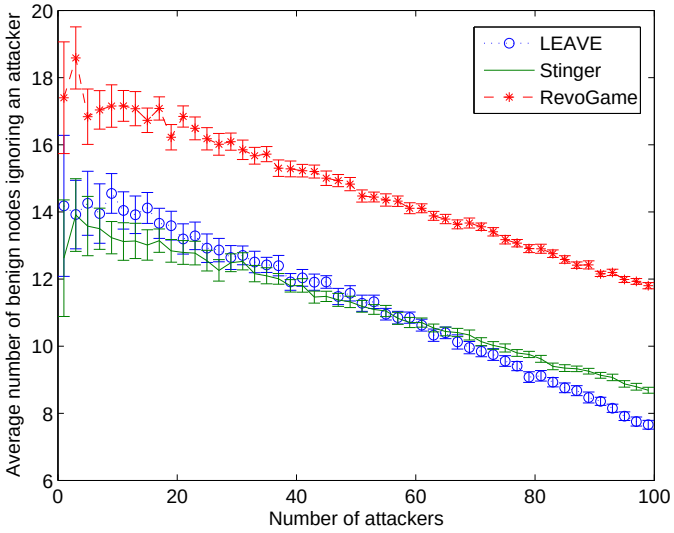


Figure 4.6: Average number of benign nodes ignoring every attacker

Similarly to the previous result, we have also calculated the *average number of benign nodes ignoring every benign node*, which is shown in Figure 4.7. On the one hand, we unsurprisingly see that Stinger's performance is the worst, analogously to the results in Figure 4.3. It is more interesting to analyze, on the other hand, the evolution of the results with LEAVE and RevoGame. When the number of attackers is small, LEAVE revokes virtually no benign nodes. Nevertheless, when the number of malicious nodes increases, we observe two different tendencies: with RevoGame, the number of benign nodes ignoring a benign node remains constant; with LEAVE, it quickly rises and exceeds that of RevoGame. This result confirms again that RevoGame performs remarkably well with a high number of attackers in the system.

This feature of RevoGame may prove very useful for the certification authority (CA) to revoke nodes permanently. One way to do this would be to set an eviction threshold; all devices that have been ignored by a certain number of nodes would be evicted from the network by the CA. As we observed in Figure 4.7, the number of benign nodes ignoring a benign node is virtually constant with RevoGame, so we can guarantee that no benign devices will be erroneously evicted by the CA. At the same time, as the number of good nodes ignoring every attacker is much higher, we are certain that the CA will revoke all malicious devices.

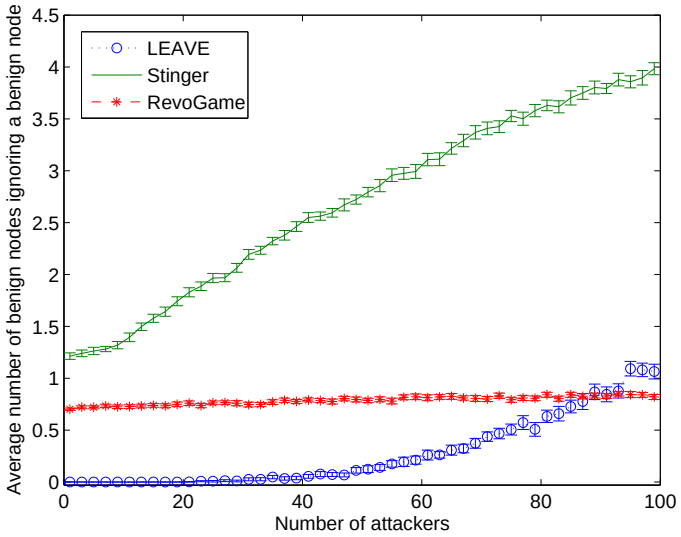


Figure 4.7: Average number of benign nodes ignoring every benign node

4.5.2 Changing the probability of detection

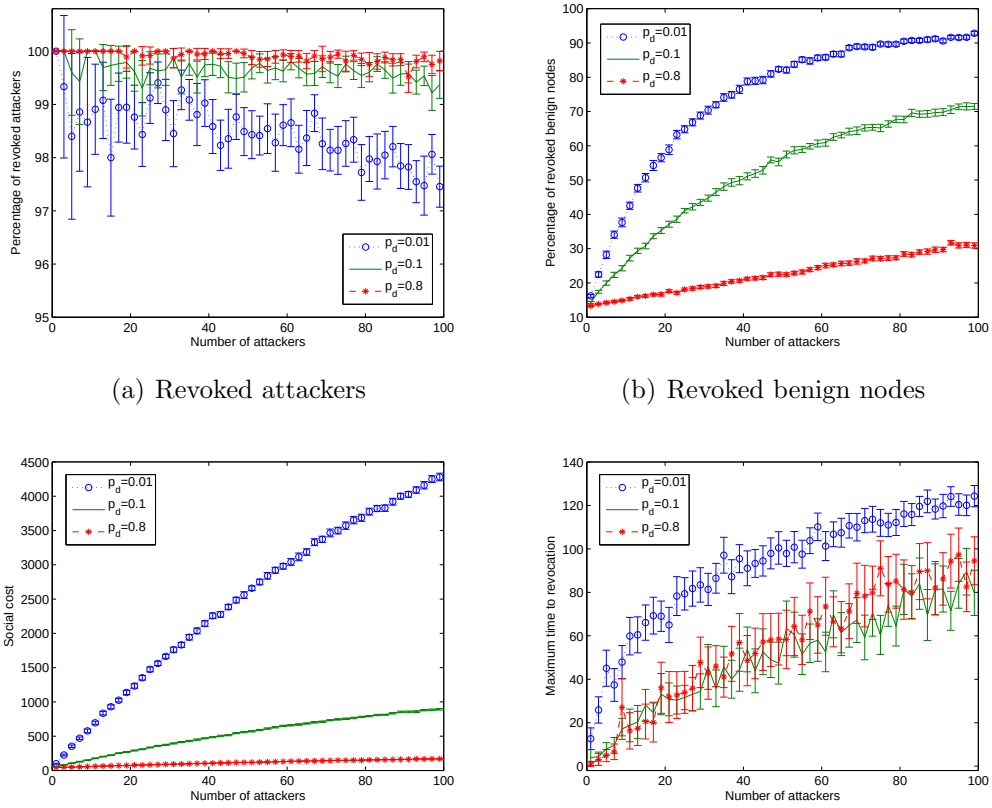
The probability of detection is a parameter with a key influence on the ability to rapidly detect and evict attackers from the network. We have performed several simulations with different p_d in order to measure its impact on the performance of RevoGame. In particular, we have set p_d to 0.1 and 0.01, and compared the obtained results with the previous ones using $p_d = 0.8$.

As it can be observed in Figure 4.8(a), the percentage of revoked attackers decreases only slightly when we reduce the probability of detection. In other words, RevoGame revokes virtually all attackers even if malicious nodes' attacks are hard to detect.

However, we see in Figure 4.8(b) a significant difference in the percentage of revoked benign nodes. The number of revoked good nodes greatly increases when p_d is reduced, especially if the number of attackers is high. Indeed, a low p_d makes it harder for benign nodes to evict attackers, so it becomes easier for malicious devices to abuse the system by revoking good nodes.

If we look at the social cost in Figure 4.8(c), we see the same tendency. As expected, a low p_d causes the social cost to be high, because of the difficulty in revoking attackers and the numerous revoked benign nodes.

Regarding the maximum time to revocation shown in Figure 4.8(d), we note that lowering p_d from 0.8 to 0.1 does not lead to a noticeable difference in the time it takes to evict misbehaving devices from the network. We see, however, that reducing it further to 0.01 does increase the time to revocation.



(a) Revoked attackers

(b) Revoked benign nodes

(c) Social cost

(d) Maximum time to revocation

Figure 4.8: Behavior of RevoGame with different values of p_d

For the sake of simplicity, we do not plot the graphs corresponding to the average number of benign nodes ignoring every attacker and the average number of benign nodes ignoring every benign node, as they do not provide any new insights.

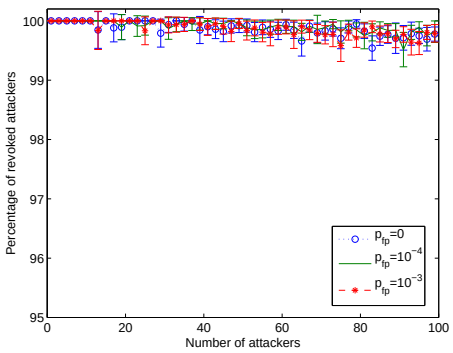
In conclusion, the results show, as expected, that a lower probability of detection indeed causes the protocol to perform worse. However, when the number of attackers is low, we might consider to reduce p_d without significantly harming the performance of RevoGame. The motivation to decrease the probability of detection is its high correlation with the probability of false positives; lowering p_d has as a side effect the reduction of p_{fp}.

4.5.3 Changing the probability of false positives

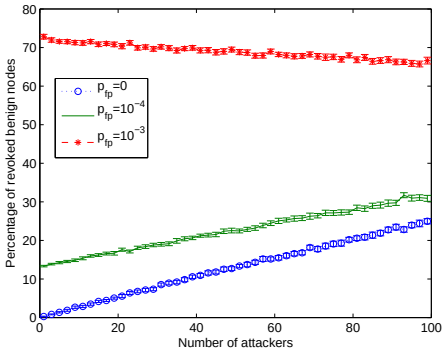
We evaluate next the behavior of the RevoGame protocol when varying the nodes' probability of false positives. In particular, we have compared the original value of $p_{fp} = 10^{-4}$ with a higher value of $p_{fp} = 10^{-3}$ and with the ideal case of $p_{fp} = 0$.

As we see in Figure 4.9(a), the probability of false positives does not affect the number of revoked attackers. However, as one would expect, p_{fp} has a huge impact on the number of revoked benign nodes; this is shown in Figure 4.9(b). In the ideal situation of $p_{fp} = 0$, all revoked good nodes are due to the malicious nodes' abuse of the system. In the other two cases, benign nodes which are evicted due to false positives add to this number. It is at first sight surprising that, in the case of $p_{fp} = 10^{-3}$, the percentage of revoked benign nodes actually drops when the number of attackers increases. The reason is that, in this case, most good nodes are revoked not by attackers, but by other benign nodes (because of the high number of false positives). As the total number of nodes in the scenario is constant, when the number of attackers increases, the number of benign nodes is reduced; therefore, there are less false positives and fewer good nodes are revoked.

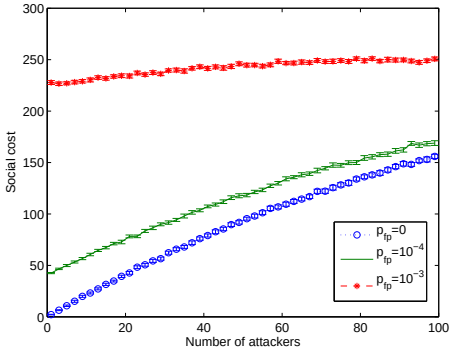
Concerning the social cost, we unsurprisingly observe in Figure 4.9(c) that it grows as p_{fp} becomes larger, due to the increase in the number of revoked benign nodes. Lastly, we do not perceive any influence of the probability of false positives on the maximum time to revocation (Figure 4.9(d)).



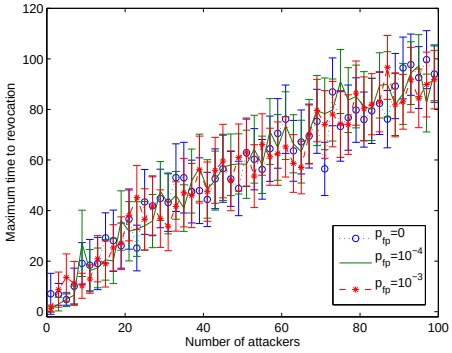
(a) Revoked attackers



(b) Revoked benign nodes



(c) Social cost



(d) Maximum time to revocation

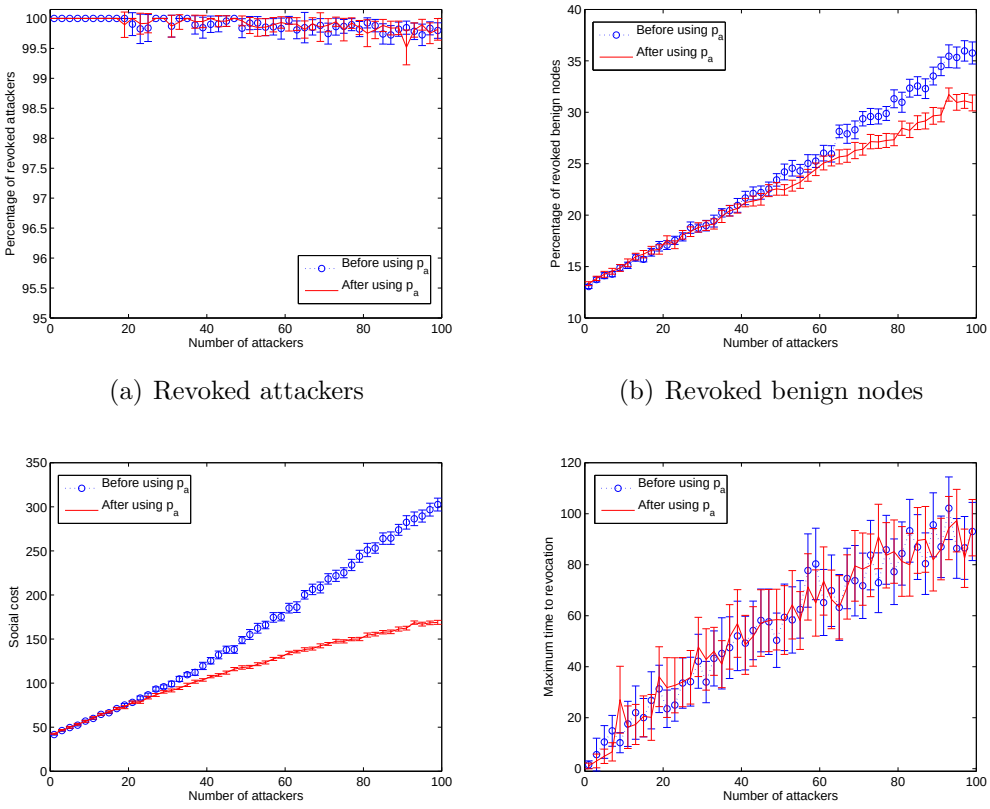
Figure 4.9: Behavior of RevoGame with different values of p_{fp}

4.5.4 Improvements made to RevoGame

Introduction of the probability of being active

In section 3.1, we discussed how to better estimate the number of players in a game thanks to p_a, the probability of being active. As we see in Figure 4.10(a), the percentage of revoked attackers did not change significantly, but we see an improvement in the number of revoked benign nodes (Figure 4.10(b)), especially when the number of attackers is large.

Regarding the social cost, Figure 4.10(c) also shows a significant diminution, caused by the decline in the number of revoked good nodes. Lastly, the time needed to evict attackers remains approximately the same (Figure 4.10(d)).



(a) Revoked attackers

(b) Revoked benign nodes

(c) Social cost

(d) Maximum time to revocation

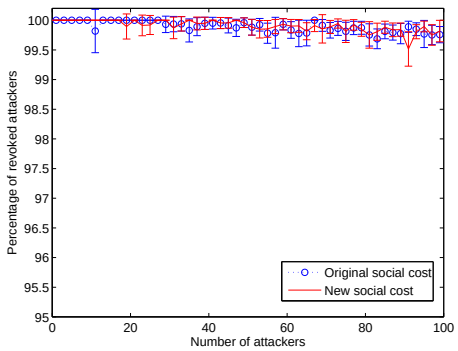
Figure 4.10: Improvement brought by the introduction of p_a

Modified social cost

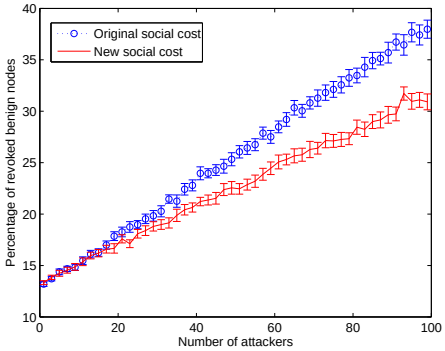
We introduced our proposed modification to the social cost of RevoGame in section 3.4. As well as its simplicity (which makes the calculation of the optimal number of votes much faster), the modified social cost also improved significantly the performance of the protocol, as we show next.

We see in Figure 4.11(b) that the protocol with the the modified social cost clearly outperforms the original scheme. Resulting from a better estimation of the optimal number of voters, less benign nodes are revoked in the simulations. The social cost (Figure 4.11(c)) is also reduced, as a consequence of the lower number of revoked good nodes.

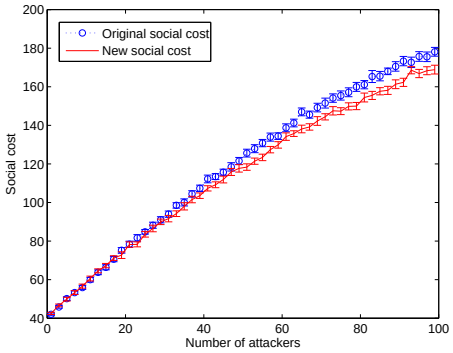
Regarding the percentage of revoked attackers (Figure 4.11(a)) and the maximum time to revocation (Figure 4.11(d)), no significant differences are observed between the original and the modified schemes.



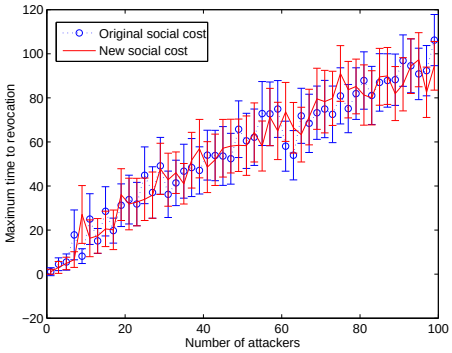
(a) Revoked attackers



(b) Revoked benign nodes



(c) Social cost



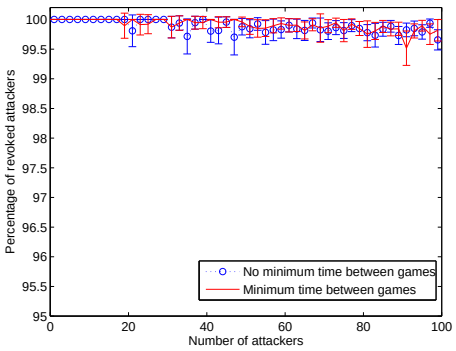
(d) Maximum time to revocation

Figure 4.11: Improvement brought by the modification of the social cost

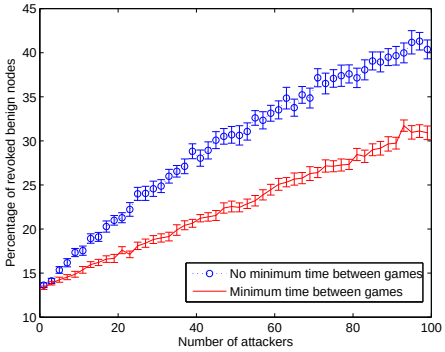
Introduction of a minimum time between games

Another of the features we incorporated to RevoGame is the minimum time between games (see section 3.6). As we can see in Figures 4.12(b) and 4.12(c), there is a clear improvement in terms of fewer revoked benign nodes and reduced social cost. The reason, as we explained in section 3.6, is that limiting the nodes' participation in revocation games minimizes the attackers' capacity to abuse the system.

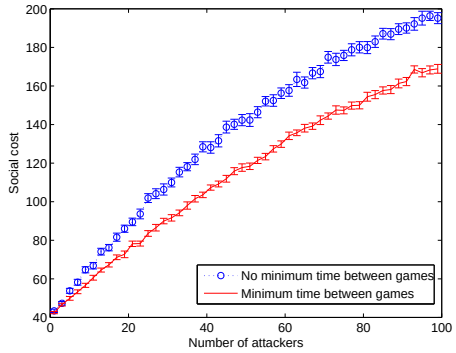
As it happened with the two previous modifications, the percentage of revoked attackers (Figure 4.12(a)) and the maximum time to revocation (Figure 4.12(d)) remain virtually unchanged after introducing the minimum time between games.



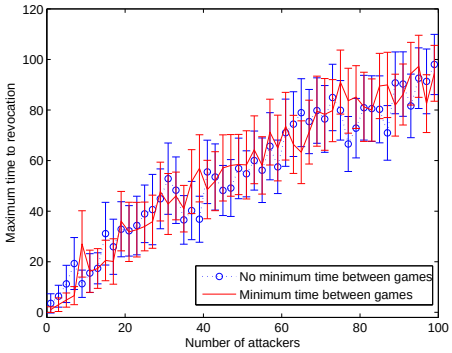
(a) Revoked attackers



(b) Revoked benign nodes



(c) Social cost



(d) Maximum time to revocation

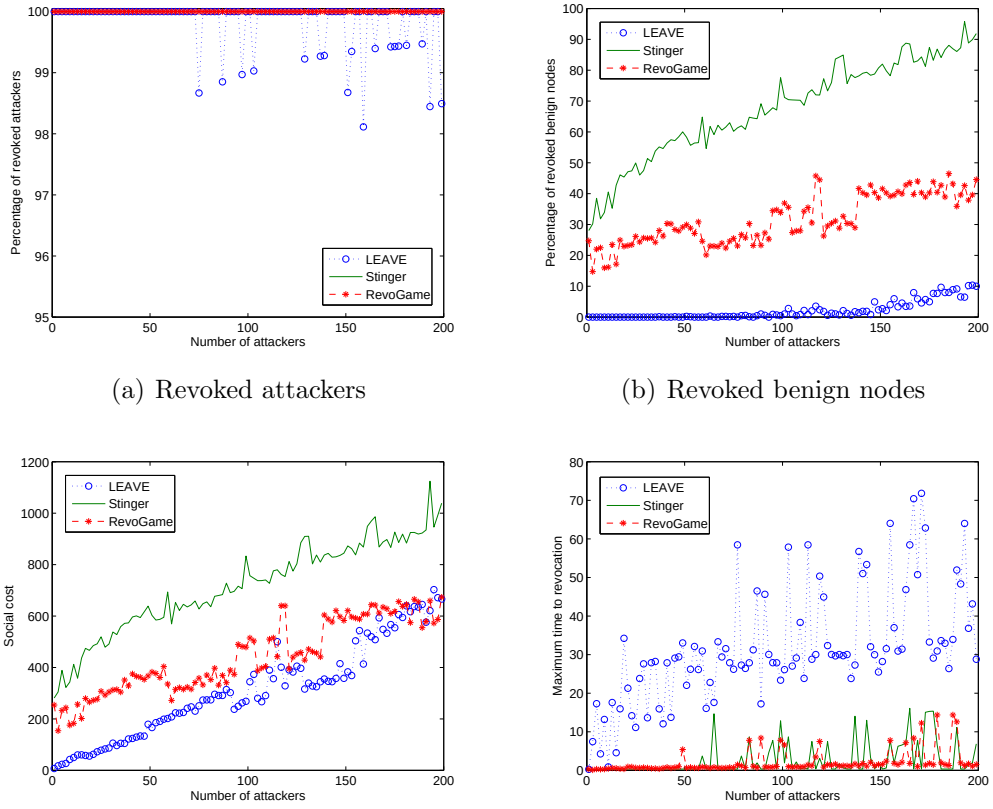
Figure 4.12: Improvement brought by the introduction of a minimum time between games

4.5.5 Scenario with a high density of vehicles

We also wanted to evaluate the performance of RevoGame in a scenario with a higher density of nodes. With this purpose, keeping the same map, we ran some simulations increasing the total number of nodes to 1000 and the maximum number of attackers to 200. We discovered that the time needed to complete the simulations increases exponentially with the number of nodes, so we had to execute the simulations without averaging.

We compared the performance of RevoGame with that of LEAVE and Stinger, as we did in section 4.5.1. We observe in Figure 4.13 that the tendencies are the same as in the scenario with 303 nodes (the variability of the results being obviously much higher due to the lack of averaging). We can

thus state that the RevoGame protocol is scalable to scenarios with a higher density of vehicles.



(a) Revoked attackers

(b) Revoked benign nodes

(c) Social cost

(d) Maximum time to revocation

Figure 4.13: Behavior of RevoGame in a high-density scenario

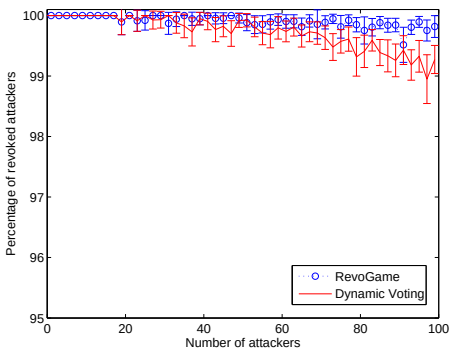
4.5.6 Dynamic Voting

We evaluate here Dynamic Voting, the alternative protocol described in section 3.7, and compare it to RevoGame. In Figure 4.14(a), we see the percentages of revoked attackers of both protocols. Dynamic Voting revokes slightly less attackers than RevoGame, because the average required number of votes to evict a malicious node from the network is higher.

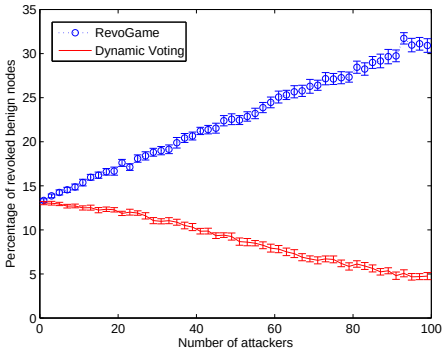
On the other hand, Dynamic Voting clearly outperforms RevoGame in terms of revoked benign nodes (Figure 4.14(b)). This huge amelioration is achieved by the increase in the required number of votes; thanks to this, the ability of attackers to abuse the system is minimized. As it happened in section 4.5.3 with $p_{fp} = 10^{-3}$, with Dynamic Voting the percentage of revoked good nodes becomes smaller when the number of attackers increases. The reason is again that most benign nodes are revoked not by attackers, but by other benign nodes. As the total number of nodes in the scenario is constant, when the

number of attackers increases, the number of benign nodes is reduced; therefore, there are less false positives and fewer good nodes are revoked.

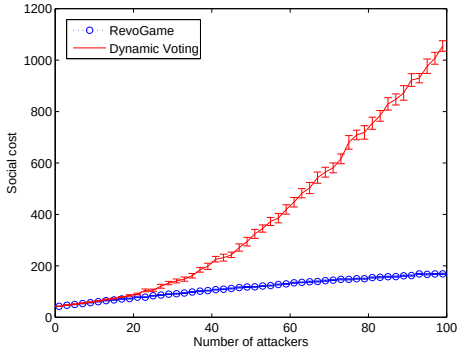
However, as the required number of votes to revoke a malicious node increases with the number of attackers, we observe in Figure 4.14(c) that the social cost of Dynamic Voting becomes very high as the number of attackers in the scenario grows. This is due to the high number of votes that are sent to the network in order to revoke all the attackers. We recall that the social cost is computed using Equation 4.1; it may be reduced by choosing a smaller value of v. Finally, the maximum time to revocation, shown in Figure 4.14(d), is the same with both protocols.



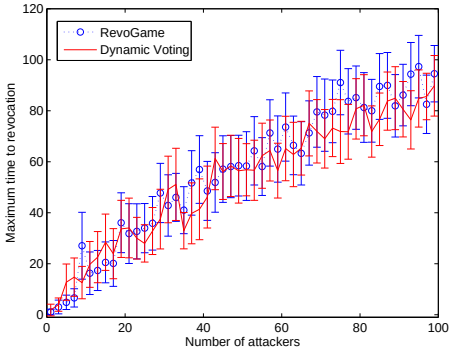
(a) Revoked attackers



(b) Revoked benign nodes



(c) Social cost



(d) Maximum time to revocation

Figure 4.14: Behavior of Dynamic Voting compared to RevoGame

Chapter 5

Conclusions

The main objective of this project was to evaluate RevoGame, a revocation protocol for ephemeral networks, by means of simulation. Based on the observed results, we have also redesigned some aspects of the protocol in order to optimize its performance. We introduced, for instance, the probability of a node being active in section 3.2, a new parameter to better estimate the number of players in a game. We also derived a new formula for the social cost, which led to the unexpected finding that most games have just one player, as explained in section 3.4. Moreover, we enforced a minimum time between games (see section 3.6) which helped to significantly reduce the number of revoked benign nodes.

We can draw several conclusions from this project. On the one hand, we have shown that RevoGame outperforms the previously proposed protocols for certificate revocation in vehicular networks LEAVE and Stinger (see section 4.5.1). On the other hand, we have also found that RevoGame might not be optimal for vehicular networks: in section 3.7, we outline Dynamic Voting, a new protocol which yields better results in the chosen scenario.

We have identified two reasons as the cause of the non-optimality of the RevoGame protocol. The first one is the existence of false positives; during a game, a node does not take them into account when deciding its strategy. The aim of RevoGame is to evict the attackers at all costs, and this causes the revocation of benign nodes when a false positive occurs.

The second reason is that the simulated network, with the parameters and the scenario chosen, cannot be considered ephemeral. For example, if two vehicles pass each other, going in opposite directions, they will be in range for 10 seconds independently of their speed (let us recall that, according to DSRC, the communication range is equal to the distance the vehicle can cover in 10 seconds at its current speed). Considering that we have set the interval between regular broadcasts to 300 ms, in compliance with the DSRC specifications, we find out that the vehicles will exchange more than 30 messages before

moving out of range. Thus, we cannot say that the interactions between nodes are ephemeral in this scenario. As RevoGame was designed for ephemeral networks, some of the assumptions made during its design do not hold in this setting, and hence the behavior of the protocol differs significantly from the expected performance.

5.1 Further work

Clearly further efforts are needed in the quest of the optimal revocation protocol for vehicular networks. In this direction, schemes based on voting with a dynamic number of votes (such as the one proposed in section 3.7) are a promising approach to overcome the drawbacks of existing protocols.

It will be also interesting to evaluate the performance of RevoGame in a truly ephemeral network. We expect it to perform significantly better in such a setting than in the tested scenario.

Bibliography

[1] `http://www.car-2-car.org`.

[2] `http://www.cs.wisc.edu/condor`.

[3] Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems — 5 GHz Band Dedicated Short Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Specifications. ASTM E2213-03, 2003.

[4] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, 1991.

[5] T. Moore, M. Raya, J. Clulow, P. Papadimitratos, R. Anderson, and J.-P. Hubaux. Fast exclusion of errant devices from vehicular networks. In *IEEE SECON*, 2008.

[6] M. Raya, H. Manshaei, M. Felegyhazi, and J.-P. Hubaux. Revocation games in ephemeral networks. In *ACM CCS*, 2008.

[7] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux. Eviction of misbehaving and faulty nodes in vehicular networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Vehicular Networks*, 25(8), 2007.

[8] V. Taliwal, D. Jiang, H. Mangold, C. Chen, and R. Sengupta. Empirical determination of channel characteristics for DSRC vehicle-to-vehicle communication. In *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 88–88, 2004.

[9] `http://trans.epfl.ch`.

[10] Intelligent Transportation Systems Standards Fact Sheet IEEE 1609 — Family of Standards for Wireless Access in Vehicular Environments (WAVE).

[11] S. Yi and R. Kravets. MOCA: Mobile certificate authority for wireless ad hoc networks. In *Proceedings of the 2nd Annual PKI Research Workshop (PKI03)*, 2003.