Revocation Games in Ephemeral Networks

Maxim Raya[†], Mohammad Hossein Manshaei[†], Márk Félegyházi[‡], Jean-Pierre Hubaux[†]

[†] School of Computer and Communication Sciences, EPFL, Switzerland
{maxim.raya, hossein.manshaei, jean-pierre.hubaux}@epfl.ch
[‡] UC Berkeley, USA
felegyha@eecs.berkeley.edu

ABSTRACT

A frequently proposed solution to node misbehavior in mobile ad hoc networks is to use reputation systems. But in ephemeral networks - a new breed of mobile networks where contact times between nodes are short and neighbors change frequently - reputations are hard to build. In this case, *local revocation* is a faster and more efficient alternative. In this paper, we define a game-theoretic model to analyze the various local revocation strategies. We establish and prove the conditions leading to subgame-perfect equilibria. We also derive the optimal parameters for voting-based schemes. Then we design a protocol based on our analysis and the practical aspects that cannot be captured in the model. With realistic simulations on ephemeral networks we compare the performance and economic costs of the different techniques.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection (e.g., firewalls)*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Algorithms, Design, Economics, Security, Theory

Keywords

Ephemeral networks, Game theory, Revocation

1. INTRODUCTION

The introduction of mobile ad hoc networks has brought numerous challenges at every layer of the protocol stack. Like others, security researchers had to solve a new set of problems such as self-organized key management [16] and misbehavior detection [27] in a decentralized, mobile, and infrastructureless environment. As ad hoc networks are typically associated with multihop routing, attacks on the routing and packet forwarding primitives have received particular attention [35]. Reputation systems, with their inherent adaptivity to a constantly changing environment, have been typically proposed as a good cure to many ad hoc networking ills [11].

Since the early days of ad hoc networks, the research landscape has seriously changed. Mostly static sensor networks, with a single owner and limited network and security infrastructure, became a focal topic of research. But packet forwarding remained an important networking component and reputation systems proved again to be effective [21]. Now, the landscape is changing again with the advancement of new types of networks, such as vehicular [34] and delay-tolerant [19] networks. The common properties of these networks are their large scale and the high mobility of wireless devices. We refer to such networks as *ephemeral networks* due to the shortness of the interactions between the participating wireless devices. In ephemeral networks, mobility makes the monitoring of neighbors' misbehavior infeasible [12]. In addition, the range of misbehavior types has extended beyond routing and packet forwarding to more diverse problems such as malicious data in vehicular networks [22]. Because traditional reputation systems are tightly coupled to a specific misbehavior type (as in packet forwarding), they cannot be merely transposed to these new types of networks. This leads us to the conclusion that new primitives are needed to replace reputation systems, where the latter cannot be efficient anymore.

In this paper, we advocate a different approach to handling misbehavior. First, the detection system should be decoupled from the reaction system. Second, the reaction should be fast and clear-cut to ensure that misbehaving nodes are denounced to their neighbors and punished despite the changing environment. Where reputation systems cannot again be the magic pill, another primitive - *local revocation* - comes to the rescue. In fact, revocation has long been part of the key management bundle of protocols [16, 17, 24, 36] and was used to cope with node compromise and misbehavior, sometimes similarly to reputation systems [5]. And although both can achieve the same goal of removing attackers from a system for certain misbehavior types, several differences exist between them.

Reputation systems, on one hand, limit the effect of attackers without specifically removing them. This means that attackers with bad reputations are naturally excluded from the network but they can redeem their good reputation by behaving correctly and benefiting from the short memory of their neighbors [10]. This also implies that monitoring nodes have to continuously run a watchdog-like mechanism [26] and keep state of their neighbors for the duration of their mutual interactions. This duration is often too short in ephemeral networks to correctly and timely react to attackers. On the other hand, revocation systems aim to remove attackers from the system and the only way revoked nodes can rejoin the system is by using new identities or credentials. Hence, the duration of the interaction and amount of state corresponding to a given attacker are limited by the time it takes to make a revocation decision.

It is worth noting that revocation typically refers to the annulment of credentials, corresponding to a compromised key, by the key issuer. The key issuer can be a Certificate Authority (CA) in managed systems [37] or the key owner herself in self-organized systems [16]. In this work, we consider revocation as a means for nodes to cope with their misbehaving neighbors, like in reputation systems. This kind of local revocation can be particularly useful when the key issuer is not available to revoke the misbehaving node (e.g., when the CA is offline) or when it is unaware of the misbehavior, especially when the latter does not involve key compromise (e.g., sending bogus information). The node that carries out the local revocation reports its result to the key issuer once the latter is reachable. A node is completely denied participation in the network only when the key issuer revokes it, based on a decision process that is out of the scope of this paper. This means that local revocation is only temporary and can be repudiated, thus preventing false revocations, due to abusers or errors, from permanently removing benign nodes from the network.

Several papers propose local revocation mechanisms as defined above. There are three major techniques that emerge: (i) voting with a fixed number of votes [17, 24, 36] or a fixed fraction of nodes (e.g., a majority) [5], (ii) key expiration and update [16], (iii) and suicide [28] whereby an accusing node can revoke an accused node by invalidating the credentials of both nodes (the high cost of revocation is meant to deter the abuse of this mechanism).[1] But what is clearly lacking is a unifying framework for comparing the various techniques and consequently defining optimum strategies for given scenarios. For example, the choice of the number of votes varies significantly among different proposals (it is 5 in [24] and sometimes more than 50 in [5]).

An extensive comparison by simulations of a voting scheme based on [5] and a suicide scheme based on [28] was carried out in [29] and gave valuable insights into the strengths and weaknesses of these two techniques in terms of security and networking performance; in addition, a heuristic hybrid protocol was proposed but not evaluated. In this paper, we make the first attempt to define an analytical framework, based on game theory, that takes into account the economic considerations of individual nodes (actually their owners, as explained below) and prove the conditions under which each strategy (namely, voting or suicide) performs best. Hence, whereas the study in [29] looks at the optimal conditions for using each strategy from a network perspective, our work focuses on the individual preferences of the nodes and then optimizes the network accordingly. In addition, our analytical framework helped us to design a protocol, RevoGame, that outperforms the other strategies.

[1]A similar strategy of expensive punishment was first introduced as a reputation system for networks of mix cascades in [18]; in this case, a single node can declare the failure of its whole mix cascade.

We chose game theory to be our modeling tool because nodes in ephemeral networks, as defined here, will belong to individuals and hence should represent the selfish nature of their owners. As we will further explain in Section 2.1, the identities of the nodes participating in revocation are a costly resource and should not be wasted (otherwise, the node owners may have to pay the logistic costs of renewing these identities). Thus, although the mechanism of choice will not be controlled by the device operator (i.e., the user), it will be implemented using a design policy palatable to the users. In fact, most users will prefer avoiding the contribution to the system (by revoking attackers) while still benefiting from its services (removing attackers benefits everyone), notably due to the potential costs of the revocation procedure. This situation is famously known as the *free rider problem* in game theory [20]. And as game theory is known to be hard to apply "as is" to networks [25], we complement our theoretical analysis with a protocol that takes practical considerations into account and we back up the resulting design using simulations on an ephemeral network.

This paper is organized as follows. In Section 2 we introduce the system model, which we follow up by the game theoretic model in Section 3 and its analysis in Section 4. In Section 5 we describe the practical protocol design and show the corresponding performance in simulations. In Section 6 we conclude the paper.

Other Related Work

The bounds on the number of voters in local majority voting have been analyzed using graph theory [30]. We do not make any explicit assumptions on the percentage of attackers. In fact, our paper tackles the revocation problem from an economic perspective, a direction that is gaining momentum in the security community [4].

Recently, cryptographers started applying game theory to multi-party computation (e.g., secret sharing). A survey of the latest results, in addition to a brief tutorial on game theory, can be found in [23]. A treatment of malicious and selfish behavior in wireless networks can be found in [13].

2. SYSTEM MODEL

2.1 Network Model

We study a network where participating nodes only have *ephemeral* (short-lived) connections, e.g., due to high mobility. In this paper we investigate Vehicular Ad hoc NETworks (VANETs), a typical example of ephemeral networks. We align our system model with the common assumption in VANETs that the nodes (i.e., vehicles and base stations) are computationally powerful, hence they can support public-key cryptography. In addition, each node contains a tamper-proof device that digitally signs every communication message [3, 31]. We assume that a CA pre-establishes the credentials for the devices offline (e.g., during the vehicle registration process), but we do not assume its presence during the execution of the local revocation protocol.

Privacy is a major concern in future wireless communication systems. To ensure location privacy for VANETs, both industry and academia have proposed the use of changing pseudonyms [3, 31]; pseudonyms can actually be public keys certified by the CA for determining liabilities in case of accidents. In line with this widely-accepted proposal, we also assume that each device changes its pseudonym periodically to avoid being tracked. Hence, a node identity is only temporary and can be renewed if the node is revoked. But, given the cost of pseudonym generation (in the logistic and not computational sense) and management by the CA, each vehicle has a limited number of pseudonyms that can quickly become a scarce resource if frequently changed.

We consider a wireless contention-based broadcast medium. For example, the medium access control protocol for VANETs will be based on the IEEE 802.11p standard. Vehicles will periodically broadcast beacon messages (e.g., every 100 or 300 ms over a range of 300 m on highways) containing the sender's pseudonym, position, speed, direction, and other safety-relevant information. The cryptographic overhead of this stream of messages is heavy but there is ongoing work to improve its efficiency [14, 15].

2.2 Adversary Model

We assume that the adversaries have the same communication capabilities as the mobile nodes. Hence, an adversary possesses the same credentials as any benign node. In this work, we consider adversaries that disseminate *false information* in the system. For example, in VANETs this false information can be a warning of the presence of ice on the road, although there is none; this may cause vehicles to make a detour to an ice-free road, thus clearing the main road for an adversary. The adversaries can be misbehaving nodes themselves or compromised benign nodes (e.g., the sensors of a vehicle are not protected from tampering and thus are exposed to attacks). We also assume that adversaries can collude, for example, by disseminating the same false information in order to increase its credibility. Last but not least, the adversaries can renew their pseudonyms, as any benign device, at the end of a pre-defined time interval (e.g., during the periodic vehicle control) and, if previously revoked, potentially restart their malicious activity.

2.3 Detection System

We assume that some of the nodes are equipped with a detection mechanism to identify false information. In VANETs, vehicles could rely on neighborhood information to verify whether there is indeed ice on the road [32]. Let p_d be the average probability with which a benign node detects an attacker. One can also interpret p_d as the fraction of nodes that possess the detection capabilities (e.g., luxury vehicles). In reality, this probability of detection depends on the nature of the false information and hence is specific to each attack. The estimation of p_d can be done in several ways. For example, a bit can be set in packets to indicate the presence of the detection equipment (e.g., a GPS device to correctly detect attacks related to position). p_d can also be the fraction of nodes observing the same event (this can be deduced based on their messages) or it can be the amount of misbehavior. Each node estimates p_d independently, but as explained in Section 5.2, our model applies even in the presence of only one detector in the system.

3. REVOCATION GAME

In this section, we introduce our game-theoretic model. The key point of the game-theoretic analysis is to consider costs when making a revocation decision. In fact, many security protocols proposed in the literature are often evaluated by their capability to cope with or to completely remove attackers. In addition, the effects of compromise and false positives corresponding to a security protocol are frequently taken into account. In this paper, we choose a different metric, namely cost, to design and evaluate a security protocol. After all, if an attack is mild (e.g., the attacker infrequently broadcasts false information), there may be no need to revoke the attacker, given the effort required to carry out the revocation. We also take into account the cost of abuse of the revocation scheme by attackers. In Section 3.1, we describe the different revocation strategies that nodes can follow. We introduce the game-theoretic model in Section 3.2 and the costs in Section 3.3. Section 3.4 is a brief overview of the game-theoretic concepts that we use in this paper.

3.1 Revocation Strategies

We consider three revocation strategies for each player (i.e., node) based on the existing protocols. First, player i can *abstain* from the local revocation procedure by playing A. This strategy assumes that player i (the index i indicates the sequence of play, which is in turn determined by the contention on the wireless channel) is not willing to contribute to the local revocation procedure and instead expects other players or eventually the CA to revoke the attacker. Second, player i can participate in a local *voting* procedure by casting a vote V against a detected attacker [17]. We assume that n votes are required to revoke an attacker locally. The choice of the value of n is a key issue in voting mechanisms and hence we optimize it in Section 4.3. Finally, following the protocol suggested in [28], we allow player i to *self-sacrifice* (denoted by the decision S), i.e., to declare the invalidity of both its current identity (the pseudonym it currently uses) and the identity of the attacker. A strategic attacker may optimize the usage of these strategies to revoke benign nodes, but we leave this attacker model for future work.

3.2 Game-Theoretic Model

We model the revocation problem using a finite *dynamic (sequential) game* **G** with wireless devices as players. Our choice of dynamic games [20] is based on the sequential nature of the wireless channel access where the action of one player is conditioned by the action of the preceding player (i.e., the second player observes, before making its decision, the action of the first player). We can represent dynamic games by their extensive form, similar to a tree where branches represent the available actions for a given player. Each level of the tree represents a *stage* of the game. We define a *revocation game* for each accused node and we assume that if several nodes are accused then there exist as many revocation games running in parallel. For the sake of analysis, we consider a single revocation game. Nonetheless, we take the parallel revocation games into account by calculating the collusion cost of the attackers playing revocation games against benign nodes.

There are N benign nodes in one game and M attackers in total; we describe how N and M are estimated in practice in Section 5.2. We assume that the number of nodes with detection capabilities is defined by the probability of detection p_d introduced in Section 2.3, i.e., there are p_dN detectors. Hence, we consider these detectors as players. Note

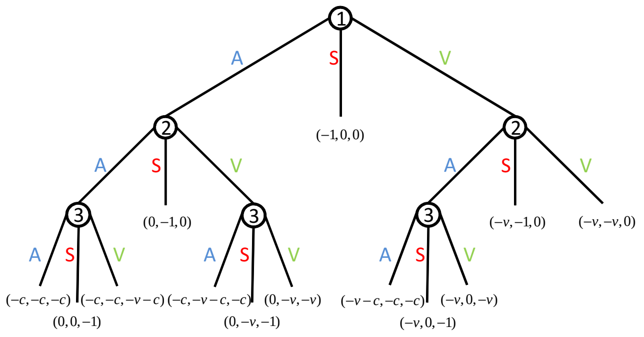


Figure 1: The extensive form of the revocation game model when the cost induced by the attack is fixed, i.e., c. The game is represented by a tree and node 1 plays the first action. The game has three stages corresponding to the moves of the three players. The actions (abstain A, self-sacrifice S, and voting V) are represented on each branch of the tree. The leaves of the tree represent the costs of the game for all players. v and 1 are the costs of voting and self-sacrifice, respectively.

that we take the non-detecting nodes also into account when calculating the social cost introduced in the next section.

As mentioned earlier, we can represent a dynamic revocation game in an extensive-form tree. We show an example in Figure 1 for $p_d N = 3$ players that want to revoke one attacker, all in power range of each other. We assume that $n = 2$ votes or a single self-sacrifice are enough to locally revoke the attacker. Due to the random nature of the medium access protocol, the sequence of the players' moves is not defined in advance; hence, we refer to the players by their respective order of successful transmission on the shared channel (i.e., index i).

3.3 Costs

We represent the costs of the players on the leaves of the extensive-form tree. The cost for any player i has two components: the cost induced by the attack and the cost of participation in revoking the attacker. All costs are represented in terms of *keys* (more precisely, pseudonyms) because they are a scarce resource, as mentioned in Section 2.1.

As shown in Figure 1, we first assume that the attack-induced cost is a constant and denote it by c. The value of this cost depends on the effect of the false information distributed by the attacker, hence we consider various values for it in Section 4. Later, we make the more plausible assumption that the attack-induced cost is variable and actually increases with time (this is especially true in time-critical safety applications in VANETs). As for the revocation costs, it is straightforward to assume that abstaining from the game does not cost the players anything. If player i sacrifices itself, then this action implies a cost of 1 key (the player totally loses its ability to use the sacrificed pseudonym). We assume that casting a vote imposes a cost v on any player i; for example, each player may have a voting quota that is decremented each time a player votes. Although voting costs in terms of computation and communication overhead, we neglect these costs in our model as we assume that nodes (vehicles in the case of VANETs) are powerful enough to justify this assumption. In contrast, the cost of pseudonym management is largely independent of the computational capabilities of the nodes for which the pseudonyms are used. We also make the reasonable assumption that $v < 1$.

For example, in Figure 1, if the three players play (A, V, V), then they successfully revoke the attacker and the second and third players bear the cost of this revocation, resulting in the costs $(0, -v, -v)$. The objective of the players is to play the strategy that minimizes their individual costs.

In addition, we introduce the notion of the *social cost* C that represents the total cost induced by attackers for all benign nodes in the system; C includes all the costs that are not captured by the sum of individual costs. Hence, given that their individual costs are minimized, nodes should also minimize the social cost, if possible (i.e., as long as minimizing the social cost does not increase the minimized individual costs). C can be expressed by:

$$C = C_{rev} + C_{abuse} + C_{fp} + C_{fn} \tag{1}$$

C_{rev} is the cost of revocation games and is equal to the sum of individual costs, C_{abuse} expresses the effect of attackers abusing the revocation system, C_{fp} is the cost of false positives, caused by incorrectly detecting benign nodes as attackers, and C_{fn} quantifies the cost of undetected attackers (false negatives).

3.4 Preliminaries

Here, we introduce some basic game-theoretic notions we use to solve the revocation problem. For the sake of simplicity, we only give an intuition for each concept. For the precise mathematical definitions we refer the reader to [20]. A brief tutorial on game theory can also be found in [13].

To predict the outcome of the extensive-form game $\mathbf{G}$, one can use the well-known concept of Nash equilibrium: A strategy profile constitutes a Nash equilibrium if none of the players can increase her payoff by unilaterally changing her strategy. Unfortunately, the Nash equilibrium concept is somewhat limited when it is applied to extensive-form games because it sometimes predicts outcomes that are not *credible* for some players (i.e., these outcomes are unreachable because the players will not play, out of self-interest, according to the incredible Nash equilibrium path). Hence, we use the stronger concept of *subgame-perfect equilibrium*. The strategy profile s is a subgame-perfect equilibrium of a finite extensive-form game $\mathbf{G}$ if it is a Nash equilibrium of any subgame $\mathbf{G}'$ (defined by the appropriate subtree) of the original game $\mathbf{G}$. "Finite game" means that the game has a finite number of stages.

One can check the existence of subgame-perfect equilibria by applying the *one-deviation property*. This property requires that there exists no single stage in the game, in which a player i can gain by deviating from her subgame-perfect equilibrium strategy while conforming to it in other stages. Hence, we can state that strategy profile s is a subgame-perfect equilibrium of a finite extensive-form game $\mathbf{G}$ if the one-deviation property holds.

We will check the existence of subgame-perfect equilibria in the revocation game by the technique of *backward induction* (also called Zermelo's algorithm in dynamic programming). Backward induction works by eliminating sub-optimal actions (i.e., yielding higher costs than the other actions in the same subtree and at the same stage of the game tree), beginning at the leaves of the extensive-form tree. The obtained path (sequence of actions) in the game tree defines the backward induction solution and any strategy profile that realizes this solution is a subgame-perfect equilibrium [20].

4. ANALYSIS

In this section, we study two types of revocation games. First, we consider a simple version where the attack-induced cost is fixed. We show that in this model, the players tend to delegate the revocation decision to the nodes playing in the last stages. This behavior is very risky if the number of players is not precisely known or if attackers can collude to remove some players before they can vote or sacrifice themselves. To overcome this limitation, we extend the first game with the assumption of increasing costs. We show that the modified game gives incentives to the game participants to play in the early stages of the revocation procedure.

4.1 Game with Fixed Costs

We assume first that the attacker causes a fixed cost if not revoked and we model this in a *revocation game with fixed costs* $\mathbf{G}^f$. We assume that $\mathbf{G}^f$ is a game of perfect information, meaning that the players know the history of play (in practice, this history is relayed by the players, as we show in Section 5.3). Figure 1 shows a simple example of $\mathbf{G}^f$. Let us recall that in this example we have $p_dN = 3$ players that want to revoke an attacker. They need $n = 2$ votes or a self-sacrifice to succeed (in Section 4.3 we describe how to compute n such that $n \leq p_dN$). Let $n_i = p_dN - i$ be the number of remaining nodes that can participate in the revocation game after node i that plays in the i^{th} stage of the game. We also assume that n_h is the number of votes that have already been cast (i.e., history of voting). Hence $n_r = n - n_h$ is the number of remaining votes that is required to revoke the attacker by voting. Theorem 4.1 identifies the strategies that a player i should follow to achieve a subgame-perfect equilibrium in $\mathbf{G}^f$. The proof is provided in Appendix A.

THEOREM 4.1. *For any given values of n_i, n_r, v, and c, the strategy of player i that results in a subgame-perfect equilibrium is:*

$$s_i = \begin{cases} A & \textit{if} \quad \begin{array}{l}[c < v] \vee [(c > 1) \wedge (n_i \geq 1)] \\ \vee[(v < c < 1) \wedge (n_i \geq n_r)]\end{array} \\ \\ V & \textit{if} \quad (v < c < 1) \wedge (n_i = n_r - 1) \\ \\ S & \textit{if} \quad (c > 1) \wedge (n_i = 0) \end{cases}$$

Essentially, Theorem 4.1 says that as the attacker cost is fixed, the only objective of the players is to remove the attacker, but they do not care in which stage. Thus, the revocation decision is left to the last players, either by voting or by self-sacrifice, whichever induces less cost. For example, a node plays S only if the attack-induced cost is higher than the cost of self-sacrifice and this node is the last player in the game.

The solution in Theorem 4.1 is not robust to estimation errors that are due to the high mobility in ephemeral networks. In fact, some of the last players may move out of radio range, thus leaving the game before their turn to play; hence, a revocation decision cannot be reached in this case. To overcome this limitation, we propose a modified version of the revocation game considering variable costs in Section 4.2. We also consider these errors when designing our practical revocation protocol in Section 5.1 by allowing the players to reassess the game conditions in each stage.

4.2 Game with Variable Costs

As explained in Section 3.3, the attack-induced cost can increase in each stage where the attacker is not revoked. We model this situation in a revocation game with variable costs $\mathbf{G}^v$. For simplicity, let us assume that the cost at stage j can be represented by $c_j = j \cdot \delta$, where δ is equal to the cost in a single stage. In our model, c_j is a linearly increasing cost, but we can derive similar results for any increasing cost function. Figure 2 (a) shows an example of $\mathbf{G}^v$ for three players. We also assume that the cost after the final stage of the revocation game grows infinitely, i.e., $\lim_{j\to\infty} c_j = \infty$ if the attacker is not revoked. In addition, we assume that $v < \delta$. Figure 2 (b) shows the simplified version of this revocation game with the above assumptions. Theorem 4.2 identifies the strategy profile that achieves a subgame-perfect equilibrium in $\mathbf{G}^v$. The proof is provided in Appendix B.

THEOREM 4.2. *For any given values of n_i, n_r, v, and δ, the strategy of player i that results in a subgame-perfect equilibrium is:*

$$s_i = \begin{cases} A & \textit{if} \quad \begin{array}{l}[(1 \leq n_i < \min\{n_r - 1, \frac{1}{\delta}\}) \\ \wedge(v + (n_r - 1)\delta < 1)] \vee [(1 \leq n_i < \frac{1}{\delta}) \\ \wedge(v + (n_r - 1)\delta > 1)]\end{array} \\ \\ V & \textit{if} \quad (n_i \geq n_r - 1) \wedge (v + (n_r - 1)\delta < 1) \\ \\ S & \textit{otherwise} \end{cases}$$

The intuition of Theorem 4.2 is that, in contrast to the game with fixed costs, players are more concerned about quickly revoking the attacker because its cost increases with time. Hence, under some conditions, they will begin the revocation process (by voting or self-sacrifice) in the early stages of the game.

4.3 Optimal Number of Voters

According to Theorem 4.1, if $v < c < 1$ and $n_i = n_r - 1$, revocation takes place by voting in $\mathbf{G}^f$. Similarly, revocation will be performed by voting in $\mathbf{G}^v$ when $n_i \geq n_r - 1$ and $v + (n_r - 1)\delta < 1$. In this section we compute the optimal number of voters n_{opt} by minimizing a simplified version of the social cost C (Eq. 1). We derive n_{opt} only for variable cost games (as mentioned in Section 4.1 and confirmed by simulations, $\mathbf{G}^f$ cannot cope with high mobility in ephemeral networks). The simplified social cost is:

$$C = n + \frac{M}{n} \tag{2}$$

The first term, n, is the amount of time, in stages, during which the attacker can damage the network and it represents the cost of revocation C_{rev}. The second term, $\frac{M}{n}$, represents the cost of abuse C_{abuse} of the revocation scheme by M adversaries (assuming the worst case scenario when M adversaries can revoke at most $\frac{M}{n}$ benign nodes). We do

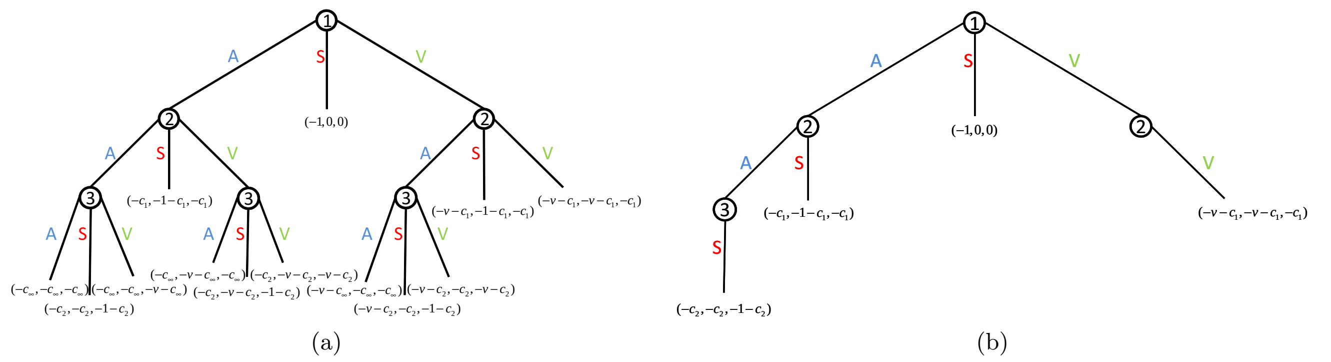


Figure 2: Game model with variable costs for three users. (a) Attack-induced costs increase in each stage j: $c_j = j \cdot \delta$ (b) Simplified version where $\lim_{j\to\infty} c_j = \infty$ and $v < \delta$. The branches corresponding to the incredible threats have been eliminated, considering the above assumptions.

not include C_{fp} and C_{fn} in the equation because their influence on the estimation of n is less significant; instead, we show by simulations their effect on the social cost. Taking into account that the number of votes should not exceed the number of players, the optimal number of votes is:

$$n_{opt} = \min\{p_d N, \sqrt{M}\} \tag{3}$$

where $n = \sqrt{M}$ is the value that minimizes C in Eq. 2. The estimation of the parameters used in the computation of n_{opt} will be explained in the next section.

5. PROTOCOLS

In this section, we describe a set of protocols that implement revocation games. Section 5.1 introduces the RevoGame protocol that selects strategies according to Theorem 4.2. We do not show the protocol for games with fixed costs because it does not work in ephemeral networks due to high mobility. Section 5.2 is a detailed evaluation of RevoGame by realistic simulations; it also compares the protocol with the voting and self-sacrifice strategies. Last but not least, section 5.3 shows how to cryptographically aggregate the votes of several players in an efficient way.

5.1 The RevoGame Protocol

The game-theoretic model presented in Section 4 allowed us to determine the optimal strategies and parameters, given some assumptions that should hold during the game. These assumptions are necessary to gain some theoretical insight into the problem, but they have to be carefully assessed when we apply the theoretical results in practical settings. For example, we derived the subgame-perfect equilibria in Section 4 by using backward induction. One of the main criticisms of backward induction in the game-theoretic community is that it relies on a long chain of assumptions on other players' actions, especially when the number of players is large. Hence, basing a strategy on the assumption that a given number of following nodes will play their predicted actions implicitly assumes that these nodes will be still in the attacker's neighborhood. But this assumption may not hold in a highly mobile network. In addition, nodes that revoke an attacker stop playing, even if they stay in the attacker's neighborhood, and hence the effective number of players decreases. The solution to these problems is twofold. First, we scale down the number of detectors $p_d N$ by a factor p_a that represents the estimated fraction of *active* players (i.e., nodes that continue playing) among the detectors. Second, the parameters used in the derivation of the optimal strategy should be re-estimated before each action.

The protocol RevoGame integrates the game-theoretic analysis in Section 4 with the practical considerations above for $\mathbf{G}^v$. In a nutshell, RevoGame estimates the different parameters, defined in Section 4, and checks the play conditions in Theorem 4.2. Based on the selected condition, RevoGame plays the corresponding strategy.

Protocol 1 RevoGame.

```
Require: $v < 1$
 1: if misbehavior detected then
 2:   $\delta \Leftarrow estimate(\delta)$
 3:   if $\delta > 1$ then
 4:     play S
 5:   else
 6:     $p_d \Leftarrow estimate(p_d)$
 7:     $p_a \Leftarrow estimate(p_a)$
 8:     $N \Leftarrow estimate(N)$
 9:     $M \Leftarrow estimate(M)$
10:     $n \Leftarrow \min\{p_a p_d N, \sqrt{M}\}$
11:     $n_r \Leftarrow n - n_h$
12:     $n_i \Leftarrow \lfloor p_a p_d N - i \rfloor$
13:     if $v + (n_r - 1)\delta < 1$ then
14:       if $n_i \geq n_r - 1$ then
15:         if $n_r = 1$ then
16:           send revocation message with vote
17:         else
18:           play V
19:       else if $1 \leq n_i < \min\{n_r - 1, \frac{1}{\delta}\}$ then
20:         play A
21:       else
22:         play S
23:     else if $1 \leq n_i < \frac{1}{\delta}$ then
24:       play A
25:     else
26:       play S
```

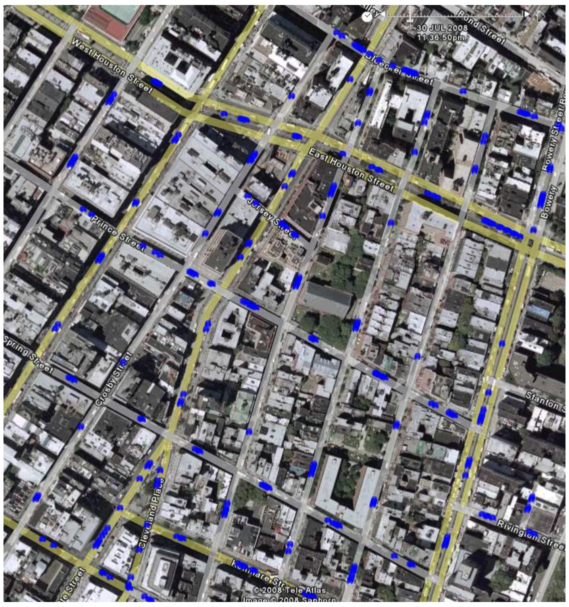


Figure 3: The simulation scenario, taken from Manhattan and rendered in Google Earth using TraNS.

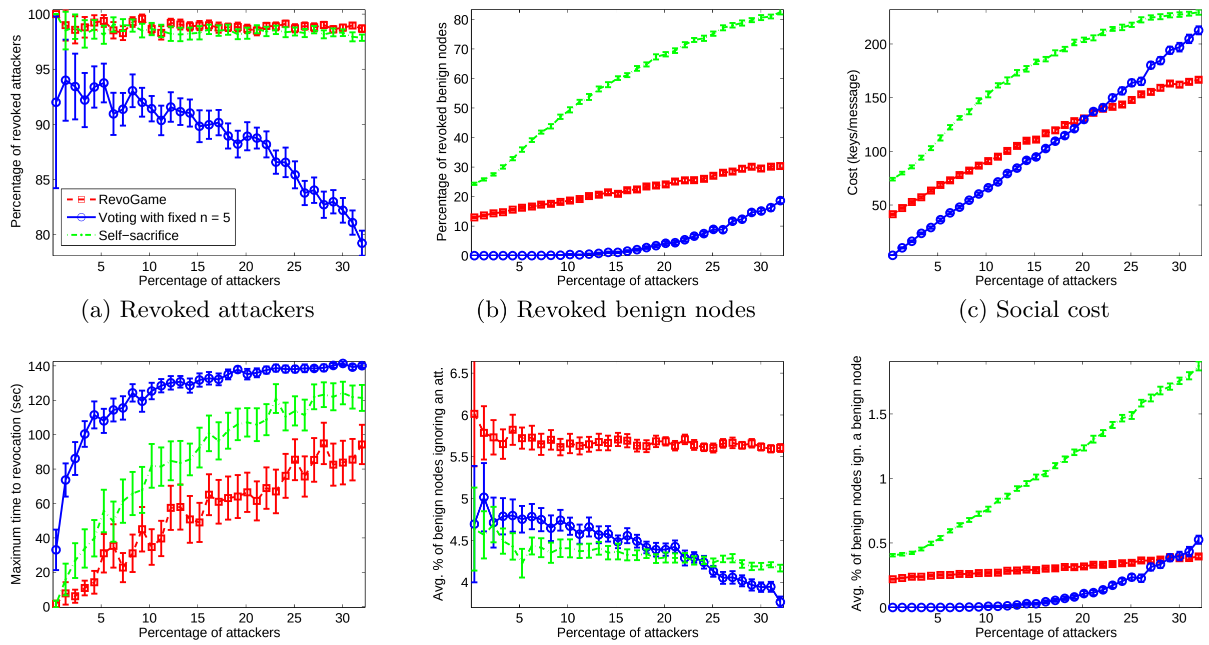
5.2 Evaluation

To evaluate our theoretical results in a practical context, we simulated the above protocol in an ephemeral network. More specifically, we focused on the re-estimation of n, given the number of attackers and detectors. To simulate the ephemeral network, we use a city scenario with 303 vehicles moving at an average speed of 50 km/h. Each vehicle broadcasts periodic messages every 300 ms over a range of approximately 150 m, conforming to the DSRC specification [2] (i.e., the communication range is the distance the vehicle can cross in 10 sec at its current speed). The scenario, illustrated in Figure 3, has an area of 6.8 km × 5.5 km in a cartesian coordinate system, is located in Manhattan, and was generated using the VANET simulation framework TraNS [1]; the underlying network simulator is ns-2. The results were averaged over 50 runs with 95 % confidence intervals. All attackers collude against benign nodes, including detectors, trying to revoke as many as possible before being revoked themselves (the attackers also cause damage by disseminating false information). We chose $p_d = 0.8$ (equivalently, 80 % of benign nodes are detectors, as explained in Section 2.3) and $\delta = 0.1$ key/message, i.e., the cost of one attacker is 0.1 key for each message sent in the network (in this case, the sequence of messages represents the time scale). Messages sent by attackers are tagged as "bad"; p_d is applied to each received message and thus represents the fact that detection is imperfect. The cost of voting v is set to 0.02. The sequential nature of the game is realized by making each node backoff a random duration between 0 and 20 ms before sending its decision, thus allowing the node to receive messages from players that have accessed the channel earlier.

To avoid any consensus-building overhead, each node has to estimate the parameters of the game independently. Hence, a node sets N to be the number of its neighbors that can hear the attacker (this can be estimated knowing the transmission ranges of nodes, in turn derived from their speeds as explained above) at the beginning of each game whereas M is the total number of attackers it has seen so far (worst-case scenario where all attackers collude). After doing a set of initial simulations, we realized that a very important parameter to properly set is p_{fp}, the probability of false positives of the detection mechanism. In our implementation, p_{fp} is the probability of identifying a message received from a benign node as "bad". Relatively high values of p_{fp} (e.g., 10^{-2}, i.e., 1 % of all messages) may result in revoking many benign nodes. Therefore, we selected a much smaller value of p_{fp}, namely 10^{-4}, which can be realized by requiring nodes to receive several "bad" messages before identifying their sender as an attacker. Last but not least, each node estimates the fraction of active players p_a by using its own history of play; more precisely, p_a is the fraction of times the node has participated in revocation games.

When we first tried to simulate the game with fixed costs, we realized that none of the attackers nor the benign nodes was revoked. This can be easily explained by the fact that in the subgame-perfect equilibrium of this game, the revocation was left to the very last players. In ephemeral networks, the assumption that these players will still be available to play does not hold because nodes move out of range very fast. This non-surprising result justifies the assumption of continuously increasing attack-induced costs (i.e., a game with variable costs). Figure 4 compares the performance of the RevoGame protocol, self-sacrifice, and voting with a fixed number of votes $n = 5$ (this is the value of n proposed in [24], the only related work where we found a specific small value for n). We can see how the game-theoretic approach adapts to the number of detectors and attackers in the system and thus performs better than the other two protocols. In fact, as this scenario is of medium density, the optimal number of votes is $n_{opt} = 1$ most of the time. Although this value may seem counterintuitive at first (it makes abuse easier), the plots actually show in detail how it surpasses the other options. It is worth noting here that a strategic adversary trying to manipulate RevoGame has little advantage as voting with 1 vote is presumably the strategy that the adversary can exploit best. In addition, $n_{opt} = 1$ shows that the algorithm for estimating n_{opt} is conservative and successfully manages to cope with varying numbers of detectors and attackers.

Figure 4(a) shows the percentage of revoked attackers. We can see that both RevoGame and self-sacrifice are efficient at revoking attackers, obviously because it is easy to revoke attackers when a revocation decision is taken by a single node; RevoGame slightly outperforms self-sacrifice because fewer detectors sacrifice themselves in the first case. The effect of the three protocols on the percentage of revoked benign nodes can be seen in Figure 4(b) where self-sacrifice removes many more benign nodes, including detectors, than RevoGame. Clearly, voting with a fixed $n = 5$ adapts poorly to a large number of attackers (because there are not enough non-revoked detectors) but avoids the revocation of benign nodes when the percentage of attackers is small. To see whether this is beneficial or detrimental to the system, we plotted the social cost of each scheme in Figure 4(c) where we can see that self-sacrifice is costlier than the other two schemes. Although RevoGame is costlier than voting for small numbers of attackers, the cost of the conservative voting scheme increases faster with the num-



(a) Revoked attackers (b) Revoked benign nodes (c) Social cost

(d) Maximum time to revocation (e) Global effect of local revocations (f) False positives and abuse

Figure 4: Performance of local revocation schemes.

ber of attackers. This is because the costs increase while the attackers remain in the system and voting takes longer to revoke attackers. Figure 4(d) shows the maximum delay between the appearance of an attacker in the system and its revocation. RevoGame performs best because it revokes fewer benign nodes than the other two schemes (put differently, it keeps more detectors in the system); as expected, voting is the slowest among the three.

Figures 4(e) and 4(f) show how the local revocation information is spread in its neighborhood; it reflects the global effect of the local mechanisms. We can see that RevoGame manages to keep the average percentage of benign nodes ignoring an attacker higher than that of the other two protocols because fewer benign nodes are revoked by RevoGame. The same reason explains why a smaller percentage of benign nodes ignore a benign node. Figure 4(f) actually gives us an interesting insight into the effect of false positives and attacker abuse of each protocol. Self-sacrifice can result in many benign nodes being ignored by other benign nodes. Voting is obviously the most resilient to false positives and abuse for relatively small percentages of attackers. But as the percentage of attackers increases, the effect of their collusion increases fast, resulting in more benign nodes being ignored. The positive aspect of RevoGame is that it keeps the average percentage of ignored benign nodes relatively stable.

In summary, RevoGame is as efficient, in terms of success rate and speed, as self-sacrifice in removing attackers but revokes fewer benign nodes, thus limiting the effect of abuse of the revocation scheme by attackers. Voting with a fixed number of votes is less efficient and slower than the other two schemes. Voting is also more resilient to the abuse of the revocation scheme when the number of attackers is relatively small, but RevoGame performs better for larger percentages of attackers. These observations justify the need for economic models of security problems, taking into account metrics (e.g., attack-induced costs) that are not captured by merely computing the number of revoked attackers and benign nodes. The protocols, such as RevoGame, built on such a model can be both efficient and adaptive to the actual security costs of a system.

5.3 Protocols for Vote Aggregation

In the case of voting, a revocation message should include the signed votes of all the voters that contributed to the revocation decision in order to limit the abuse of the revocation scheme by attackers. A straightforward approach to do this is to concatenate all the votes in the revocation message. But this would inflate both the size and the verification time of the revocation message, which is highly undesirable in ephemeral networks. Fortunately, savings in space, and sometimes time, can be achieved by using a special cryptographic construction - *aggregate signatures* - that allows for the compression of multiple signatures into a single one. Two possible constructions are suitable in our case: general and sequential aggregate signatures. In the following, we describe how these constructions can be applied to vote aggregation and compare them.

5.3.1 General Aggregate Signatures

General aggregate signatures allow any node to aggregate the signatures of other nodes in any order. For the sake of generality, we define the three algorithms GenSign, GenAggSign, and GenAggVerify to designate individual signature generation, aggregate signature generation and aggregate signature verification, respectively. Although any general aggregate signature scheme can be used, our scheme of choice in this work is the BGLS signature [8] based on the BLS short signatures [9].

Let v_i be the public key of a node i, v_a be the public key of the attacker being revoked, σ_i be the individual signature of i, σ'_i be the aggregate signature so far, n'_i be the optimal number of voters computed by i, t_i the current timestamp of i, $vote_i$ the message (vote) of i, and $vote'_i$ the aggregate vote so far. Let the field $flag_r = 1$ designate that the revocation decision is reached, otherwise $flag_r = 0$. As general aggregate signatures are unordered, a voter can precompute the signature on its vote prior to the reception of the other votes. If the revocation decision is not yet reached, the aggregate-so-far signature on the other votes σ'_{i-1} can be aggregated with σ_i (note here that the index i does not refer to the signing order but to the transmission order). If the revocation decision is reached, σ_i has to be recomputed with $flag_r$ set to 1. In addition to aggregating the votes, the certificates $cert_i$ of the public keys v_i can be aggregated as well, using the same primitive. In this case, "certificate" specifically means the signature of the CA over v_i, hence certificates can be aggregated with the signatures on votes. We assume that nodes know the public key of the CA and thus are able to verify its certificates. The protocol GenVote illustrates the vote aggregation process.

Protocol 2 GenVote

```
Ensure: $flag_r = 0$
1: $vote_i \Leftarrow v_a \| flag_r \| v_i \| n'_i \| t_i$
2: $\sigma_i \Leftarrow \mathsf{GenSign}(vote_i)$ // precompute the signature
3: if votes received then
4:   if decision = revocation then
5:     $flag_r \Leftarrow 1$
6:     $vote_i \Leftarrow v_a \| flag_r \| v_i \| n'_i \| t_i$
7:     $\sigma_i \Leftarrow \mathsf{GenSign}(vote_i)$ // recompute the sig.
8:   $\sigma'_i \Leftarrow \mathsf{GenAggSign}(\sigma_i, cert_i)$
9:   $vote'_i \Leftarrow v_a \| flag_r \| v_1 \| n'_1 \| t_1 \| \ldots \| v_i \| n'_i \| t_i \| \sigma'_i$
```

GenVoteVerify illustrates the vote verification process. This is done only if $flag_r = 1$, i.e., a revocation decision has been reached. First, the votes need to be reconstructed and then the aggregate signature is verified.

Protocol 3 GenVoteVerify($vote'_k$)

```
1: $v_a, flag_r, \sigma'_k \Leftarrow extract(vote'_k)$
Ensure: $flag_r = 1$
2: for $i = 1$ to $k$ do
3:   $v_i, n'_i, t_i \Leftarrow extract(vote'_k)$
4:   if $i < k$ then
5:     $vote_i \Leftarrow v_a \| 0 \| v_i \| n'_i \| t_i$ // reconstruct the votes
6:   else
7:     $vote_k \Leftarrow v_a \| 1 \| v_k \| n'_k \| t_k$
8: $\mathsf{GenAggVerify}(vote_1, \ldots, vote_k, v_1, \ldots, v_k, \sigma'_k)$
```

5.3.2 *Sequential Aggregate Signatures*

Although general aggregate signatures sometimes allow for saving time on signature computation, they require both the public key and the certificate of a voter to be sent over the air. A common solution to this problem is to use identity-based signatures (IBS) where the identity of a node serves as its public key and its capability to sign plays the role of the certificate in the previous scheme. The only possible, so far, aggregate constructions using IBS are sequential aggregate signatures where signatures are generated in a given order and aggregation and signing are the same operation [6]. This means that it is not possible to precompute individual signatures. The advantage of identity-based signatures is that no certificates need to be added to the signatures. Given that it is easier to optimize the computation time, as explained in the next section, rather than the message overhead, sequential aggregate signatures based on IBS can be a viable option in our scenario.

We define IBSeqSign and IBSeqVerify to designate the aggregate signature generation and verification algorithms. Let ID_i be the identity of node i and ID_a be the identity of the attacker. We assume that the aggregate signature construction is based on the IBSAS scheme recently proposed in [6]. The protocol IBSeqVote illustrates the vote aggregation process. The same definitions as in the previous section apply.

Protocol 4 IBSeqVote

```
Ensure: $flag_r = 0$
1: if $vote'_{i-1}$ received then
2:   for $j = 1$ to $i - 1$ do
3:     $ID_j, n'_j, t_j \Leftarrow extract(vote'_{i-1})$
4:   if decision = revocation then
5:     $flag_r \Leftarrow 1$
6:   $vote_i \Leftarrow ID_a \| flag_r \| ID_1 \| n'_1 \| t_1 \| \ldots \| ID_i \| n'_i \| t_i$
7:   $\sigma_i \Leftarrow \mathsf{IBSeqSign}(vote_i)$
8:   $vote'_i \Leftarrow vote_i \| \sigma_i$
```

We omit the description of IBSeqVerify as it is similar to GenVoteVerify.

5.3.3 *Comparison*

A general aggregate signature can be as short as 171 bits (for 1024-bit security) if the individual signatures are BLS short signatures [9], but requires the certificates of the public keys of all signers. Although certificates can be aggregated as described in the previous section, this may not always be possible (e.g., if certificates are not based on the BLS short signatures). An additional drawback of the general aggregate signatures based on the BGLS scheme is that the verification time requires a linear, in the number of voters, number of pairing computations [8], an expensive cryptographic primitive.

The identity-based aggregate signatures are longer than the BLS signatures but require no certificates; they still require the transmission of the identities of all the signers. An additional advantage of the IBSAS scheme is that verification of an aggregate signature requires three pairing computations independently of the number of signers [6].

In summary, in terms of communication overhead, general aggregate signatures are comparable to sequential aggregate signatures if the certificates in the first scheme are aggregated; otherwise, the latter scheme generates shorter aggregate signatures. In terms of computation overhead, general aggregate signatures are faster to generate (especially with precomputations) but slower to verify, whereas sequential aggregate signatures are slower to generate but faster to verify. As a last note on the efficiency of pairing computations, recent results have shown the feasibility of pairings on resource-constrained devices such as smart cards [33]. Of course, the platform of choice plays a crucial role in the execution speed, but it is reasonable to expect that efficient bilinear pairings will soon be a common and viable cryptographic primitive [7].

6. CONCLUSION

New types of networks usually require new security mechanisms. Ephemeral networks are an example where reputation systems may not perform well. In this paper, we have studied the applicability of local revocation mechanisms to handle misbehavior in these networks. Using a game-theoretic model, we have derived the optimal strategies and parameters for different combinations of detection capability and attacker penetration and impact. In addition, we have designed a protocol, RevoGame, based on the game-theoretic analysis and practical considerations. Realistic simulation results in vehicular networks show that the game-theoretic approach achieves the elusive tradeoff between the approaches found in the literature.

Acknowledgements

We would like to thank Tansu Alpcan, Julien Freudiger, Vern Paxson, Marcin Poturalski, Reza Shokri, Istvan Vajda, our CCS shepherd Paul Syverson, and the anonymous reviewers for their helpful feedback on earlier versions of this work. Special thanks go to Ignacio Llatser who implemented the simulations and gave us interesting insights into the project. This work was supported (in part) by the European STREP project SeVeCom (`http://www.sevecom.org`) and the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

7. REFERENCES

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

[2] 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.

[3] IEEE P1609.2 Version 1 - Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages. *In development*, 2006.

[4] R. Anderson and T. Moore. The economics of information security. *Science*, 314(5799):610–613, Oct. 2006.

[5] G. Arboit, C. Crepeau, C.R. Davis, and M. Maheswaran. A localized certificate revocation scheme for mobile ad hoc networks. *Ad Hoc Networks*, 6(1):17–31, January 2008.

[6] A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing (extended abstract). In *Proceedings of CCS'07*.

[7] D. Boneh. A brief look at pairings based cryptography. In *Proceedings of FOCS'07*.

[8] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Proceedings of EUROCRYPT'03*, volume 2656 of *LNCS*, pages 416–432.

[9] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, 2004.

[10] S. Buchegger and J.-Y. Le Boudec. A robust reputation system for P2P and mobile ad-hoc networks. In *Proceedings of P2PEcon'04*.

[11] S. Buchegger and J.-Y. Le Boudec. Self-policing mobile ad hoc networks by reputation systems. *Communications Magazine, IEEE*, 43(7):101–107, 2005.

[12] J. Burgess, G. D. Bissias, M. Corner, and B. N. Levine. Surviving attacks on disruption-tolerant networks without authentication. In *Proceedings of MobiHoc'07*.

[13] L. Buttyan and J.-P. Hubaux. *Security and Cooperation in Wireless Networks*. Cambridge University Press, 2007. `http://secowinet.epfl.ch`.

[14] G. Calandriello, P. Papadimitratos, A. Lioy, and J.-P. Hubaux. Efficient and robust pseudonymous authentication in VANET. In *Proceedings of VANET'07*.

[15] J. Camenisch, S. Hohenberger, and M. Ø. Pedersen. Batch verification of short signatures. In *Proceedings of EUROCRYPT'07*, volume 4515 of *LNCS*, pages 246–263.

[16] S. Capkun, L. Buttyan, and J.P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(1):52–64, 2003.

[17] H. Chan, V. D. Gligor, A. Perrig, and G. Muralidharan. On the distribution and revocation of cryptographic keys in sensor networks. *IEEE Transactions on Dependable and Secure Computing*, 2(3):233–247, 2005.

[18] R. Dingledine and P. Syverson. Reliable mix cascade networks through reputation. In *Proceedings of FC'02*, volume 2357 of *LNCS*, pages 253–268.

[19] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of SIGCOMM'03*.

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

[21] S. Ganeriwal and M. Srivastava. Reputation-based framework for high integrity sensor networks. In *Proceedings of SASN'04*.

[22] P. Golle, D. Greene, and J. Staddon. Detecting and correcting malicious data in VANETs. In *Proceedings of VANET'04*.

[23] J. Katz. Bridging game theory and cryptography: Recent results and future directions. In *Proceedings of TCC'08*, volume 4948 of *LNCS*, pages 251–272.

[24] H. Luo, J. Kong, P. Zerfos, S. Lu, and L. Zhang. URSA: ubiquitous and robust access control for mobile ad hoc networks. *IEEE/ACM Transactions on Networking*, 12(6):1049–1063, December 2004.

[25] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Experiences applying game theory to system design. In *Proceedings of PINS'04*, 2004.

[26] S. Marti, T.J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of MobiCom'00*.

[27] A. Mishra, K. Nadkarni, and A. Patcha. Intrusion detection in wireless ad hoc networks. *IEEE Wireless Communications*, 11(1):48–60, Feb. 2004.

[28] T. Moore, J. Clulow, S. Nagaraja, and R. Anderson.

New strategies for revocation in ad-hoc networks. In *Proceedings of ESAS'07*.

[29] T. Moore, M. Raya, J. Clulow, P. Papadimitratos, R. Anderson, and J.-P. Hubaux. Fast exclusion of errant devices from vehicular networks. In *Proceedings of SECON'08*.

[30] D. Peleg. Local majorities, coalitions and monopolies in graphs: a review. *Theoretical Computer Science*, 282(2):231–257, Jun. 2002.

[31] M. Raya and J.-P. Hubaux. The security of vehicular ad hoc networks. In *Proceedings of SASN'05*.

[32] M. Raya, P. Papadimitratos, V. Gligor, and J.-P. Hubaux. On data-centric trust establishment in ephemeral ad hoc networks. In *Proceedings of INFOCOM'08*.

[33] M. Scott, N. Costigan, and W. Abdulwahab. Implementing cryptographic pairings on smartcards. In *Proceedings of CHES'06*, volume 4249 of *LNCS*, pages 134–147.

[34] Q. Xu, T. Mak, J. Ko, and R. Sengupta. Vehicle-to-vehicle safety messaging in DSRC. In *Proceedings of VANET'04*.

[35] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang. Security in mobile ad hoc networks: challenges and solutions. *Wireless Communications, IEEE*, 11(1):38–47, 2004.

[36] S. Yi and R. Kravets. MOCA: Mobile certificate authority for wireless ad hoc networks. In *Proceedings of PKI'03*.

[37] P. Zheng. Tradeoffs in certificate revocation schemes. *SIGCOMM Computing Communication Review*, 33(2):103–112, 2003.

APPENDIX

A. PROOF OF THEOREM 4.1

Proof. We use the one-stage-deviation principle to prove that deviating from each of the strategies in Theorem 4.1 under the corresponding conditions will not result in a gain.

Let us assume that $c < v$, i.e., voting is more expensive than enduring the attack-induced cost. If at any stage, player i deviates from the strategy A, playing V or S would result in a cost of v or 1, respectively. In both cases, the cost is bigger than c (assuming that $v < 1$ as mentioned in Section 3.3). Figure 5 illustrates this case.

If $v < c < 1$ and $n_i \geq n_r$, i.e., voting is less expensive than the attack-induced cost and the number of remaining detectors is bigger than the required number of voters, then playing S or V would result in a cost of 1 or v, respectively. These costs are greater than 0 (the attacker will be revoked anyway because $v < c$). Hence, i cannot gain by deviating from the action A. This is shown in Figure 6 for $p_d N = 3$.

Another case that makes action A the best response is when the attack-induced cost c is bigger than 1, the cost of self-sacrifice, and the number of remaining players is bigger than 1 ($n_i \geq 1$), i.e., the attacker will be revoked by another player anyway. The proof is similar to the previous cases and is illustrated in Figure 7.

Let us now assume that $v < c < 1$, $n_i = n_r - 1$, and player i that is supposed to play V according to strategy s_i above, deviates in a single stage. If it plays S or A, it loses 1 or c, respectively, both bigger than v. In both cases, i cannot gain by deviating from V.

Finally, if $c > 1$ and $n_i = 0$, if player i deviates from S by playing A or V, the player's cost will be c or $c + v$, respectively. Both costs are greater than 1 and deviation results in a loss.

Based on the above cases, deviation from any action under the corresponding conditions results in a loss for the deviating player and hence strategy s_i leads to a subgame-perfect equilibrium. □

B. PROOF OF THEOREM 4.2

Proof. We will prove this theorem, as before, by showing that the actions in the theorem are the players' best responses under the corresponding conditions.

If $1 \leq n_i < \min\{n_r - 1, \frac{1}{\delta}\}$ (i.e., there are not enough voters but at least another player can self-sacrifice) and $v + (n_r - 1)\delta < 1$ (i.e., self-sacrifice is more expensive than voting), deviating from playing A will cause player i a cost of 1, because playing S is the only possible option (the number of voters is insufficient). Hence, i does not gain by deviation. This is shown in the subtree of player 2 on the left in Figure 8.

If $1 \leq n_i < \frac{1}{\delta}$ and $v + (n_r - 1)\delta > 1$, deviating from playing A will cause player i a cost of 1 if it plays S and a cost of $v + (n_r - 1)\delta$ if it plays V. Hence, i does not gain by deviation from A. This is illustrated in the subtree of player 2 on the left in Figure 9.

If $n_i \geq n_r - 1$ and $v + (n_r - 1)\delta < 1$, i.e., there are enough voters and voting is less expensive than self-sacrifice, and player i deviates from playing V by playing A or S, its cost will be $n_i\delta$ or 1, respectively. In both cases the cost will be greater than $v + (n_r - 1)\delta$, assuming v is negligible. Hence, the player does not gain by one-stage deviation. This is shown in Figure 8.

The expanded condition for playing S is: $[\delta > 1] \vee [(n_i < n_r - 1) \wedge ((n_i = 0) \vee (n_i \geq \frac{1}{\delta})) \wedge (v + (n_r - 1)\delta < 1)] \vee [((n_i = 0) \vee (n_i \geq \frac{1}{\delta})) \wedge (\delta < 1 < v + (n_r - 1)\delta)]$. If $\delta > 1$ (the attack-induced cost is more expensive than self-sacrifice) and player i deviates from strategy S, it can play V or A. If it plays V, it loses $v + (n_r - 1)\delta > 1$ and if it plays A, it loses $n_i\delta > 1$ as shown in Figure 10.

If $n_i = 0$ (the current player is the last one) or $n_i \geq \frac{1}{\delta}$ (alternately $n_i\delta \geq 1$, which means that the cost of abstaining is higher than the cost of self-sacrifice), and $v + (n_r - 1)\delta < 1$ and $n_i < n_r - 1$ (i.e., voting is cheaper than self-sacrifice but there are not enough voters), deviating from S by playing A would result in a cost of $n_i\delta$ if $n_i \geq \frac{1}{\delta}$ or ∞ if $n_i = 0$ (the attacker will not be revoked). Playing V would not lead to revocation (as there are not enough voters) and hence result in a cost of $v + n_i\delta$ if $n_i \geq \frac{1}{\delta}$ or ∞ if $n_i = 0$. Hence, deviation from S would not pay off. This is shown in the subtree of player 3 in Figure 8. A similar proof can be made if $n_i = 0$ or $n_i \geq \frac{1}{\delta}$, and $\delta < 1 < v + (n_r - 1)\delta$: both voting and abstaining are more expensive than self-sacrifice and hence deviation from S would increase the cost (in other words, lower the payoff). This can be seen in the subtree of player 3 in Figure 9. Based on the above, one-stage-deviation from the strategy s_i degrades the deviating player's payoff and hence s_i leads to a subgame-perfect equilibrium. □

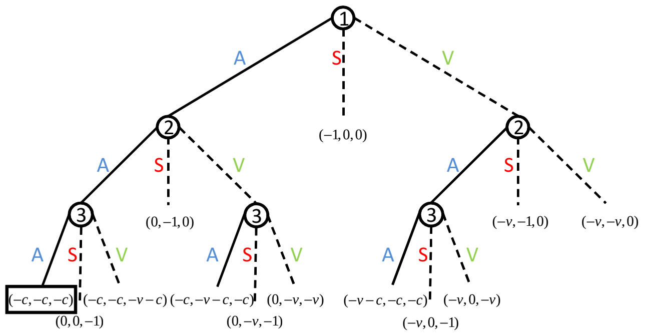


Figure 5: Extensive form of G^f when $c < v$. Action A for all players is the best response. The thick lines show the best response of each player in every stage of the game. This can be obtained by comparing the resulting costs for all strategies of a player moving in a given stage. The continuous thick line represents the subgame-perfect equilibrium (all nodes play A).

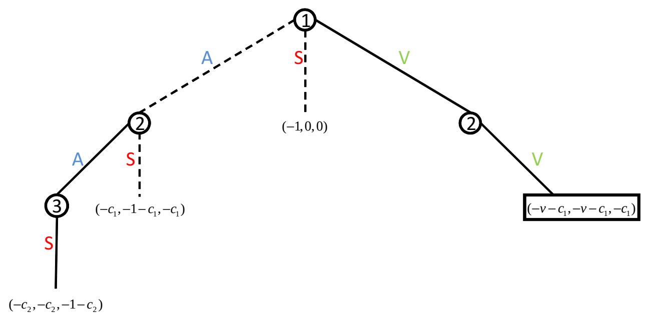


Figure 8: Extensive form of G^v when $v+(n_r-1)\delta < 1$. The subgame perfect equilibrium is achieved when players 1 and 2 play V.

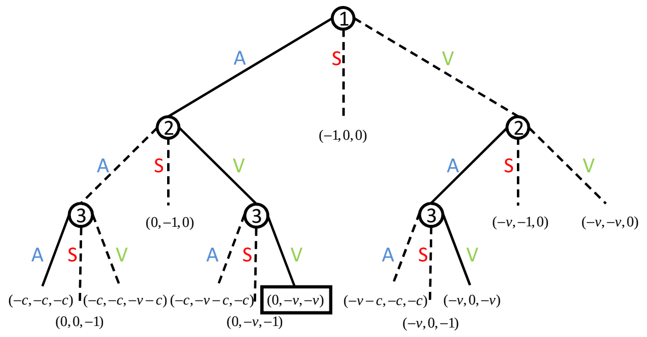


Figure 6: Extensive form of G^f when $v < c < 1$. Action A for the first player is the best response since $n_i \geq n_r$, whereas the best response for nodes 2 and 3 is V since $n_i < n_r$. Hence, the subgame-perfect equilibrium game is achieved if player 1 plays A, 2 plays V, and 3 plays V.

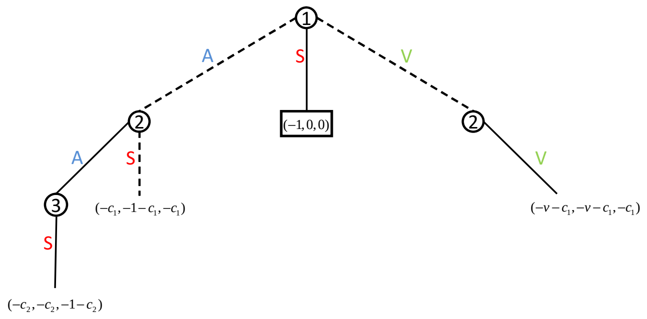


Figure 9: Extensive form of G^v when $v + (n_r - 1)\delta > 1 > \delta$. The subgame-perfect equilibrium is achieved when player 1 plays S.

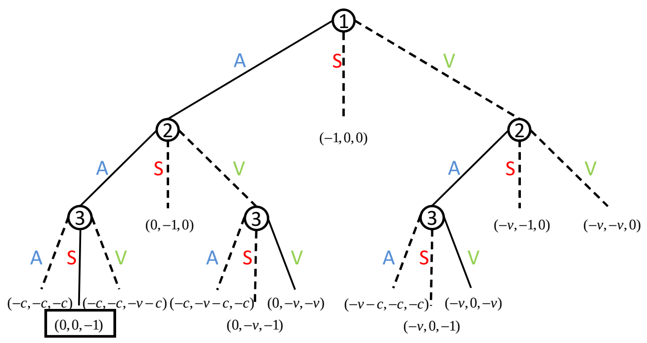


Figure 7: Extensive form of G^f when $c > 1$. Action A is the best response of the first and second players because $n_1 > n_2 \geq 1$, whereas player 3 plays S because $n_3 = 0$.

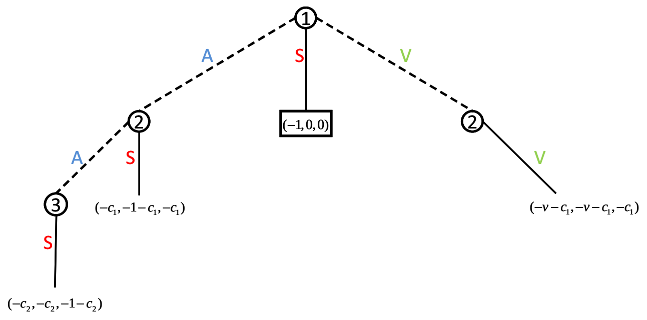


Figure 10: Extensive form of G^v when $\delta > 1$. The subgame-perfect equilibrium is achieved when player 1 plays S.