A Resilient and Energy-saving Incentive System for Resource Sharing in MANETs*

Holger Teske, Jochen Furthmüller, and Oliver P. Waldhorst

Institute of Telematics,
Karlsruhe Institute of Technology
Zirkel 2, 76131 Karlsruhe, Germany
holger.teske@student.kit.edu, furthmueller|waldhorst@kit.edu

Abstract

Despite of all progress in terms of computational power, communication bandwidth, and feature richness, limited battery capacity is the major bottleneck for using the resources of mobile devices in innovative distributed applications. Incentives are required for motivating a user to spend energy on behalf of other users and it must be ensured that providing these incentives neither consumes much energy by itself nor allows for free-riding and other types of fraud. In this paper, we present a novel incentive system that is tailored to the application scenario of energy-aware resource-sharing between mobile devices. The system has low energy consumption due to avoiding the use of public key cryptography. It uses a virtual currency with reusable coins and detects forgery and other fraud when cashing coins at an off-line broker. A prototype-based measurement study indicates the energy-efficiency of the system, while simulation studies show its resilience to fraud. Even in scenarios with 75 % of fraudulent users that are colluding to disguise their fraud only 3.2 % of them get away with it while the energy overhead (about 3 %) for the incentive system is still moderate

Keywords and phrases incentive system; resource sharing; fraud detection

Digital Object Identifier 10.4230/OASIcs.KiVS.2011.109

1 Introduction

Current sales figures[1] indicate a significant increase in the popularity of smart phones and evidently show a trend towards feature-rich mobile devices. Besides offering computing and storage resources almost comparable to desktop PCs ten years ago, such devices offer a variety of other resources, including different communications capacities like 3G, WiFi, and Bluetooth, as well as sensors for position, acceleration, light, and temperature. Combining the resources provided by multiple devices enables new and exciting applications. These are typically observed as a natural subset of pervasive computing [25] and find increasing interest in many other disciplines of distributed computing, e. g., in Grid computing [10] and service overlays [5]. Example applications range from pooling capacities of the cellular connections of multiple devices to speed up downloads [3] to people-centric sensing exploiting the sensors of thousands of smart-phones [8].

Unfortunately, despite of the growth in resource variety, processor speed, memory size, and communication bandwidth, battery capacity remains the limiting factor for realizing the vision described above [18]. Providing resources for applications running on remote devices may consume a significant amount of energy, limiting the operating time of a mobile device for the owner's personal use. In fact, mechanisms are required to motivate device owners—that are not known to each other in general and, thus, do not pursue a common goal—to spend energy on behalf of others. Such mechanisms can be provided by *incentive*

* This research is supported by the "Concept for the Future" of Karlsruhe Institute of Technology within the framework of the German Excellence Initiative.
[1] http://www.gartner.com/it/page.jsp?id=1306513

17th GI/ITG Conference on Communication in Distributed Systems (KiVS'11).
Editors: Norbert Luttenberger, Hagen Peters; pp. 109–120
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

systems. These systems could recompense the energy spent for serving a remote resource request, and allow to use the refund in turn to recompense others for using their resources.

Many incentive systems for motivating cooperation among users have been proposed with different application scenarios in mind, e. g. MilliCent [20], NetPay [9], and Micromint [24]. However, most of them cannot be used to motivate resource sharing among mobile devices, since they either require trusted hardware, connections to a central broker or other third parties on each interaction that requires a refund, or utilize refunds that cannot be reused without opening the door for fraud. An even more important drawback when it comes to providing incentives for spending energy is that most systems consume lots of energy by themselves, e. g., by requiring the use of public key cryptography on each payment, contradicting the primary goal of the incentive system.

In this paper, we present an energy-efficient and resilient incentive system tailored to the use-case of recompensing the energy required for resource sharing. For achieving energy-efficiency, the system works entirely without public key cryptography. Nevertheless, it is based on a virtual currency that can be exchanged between participants without communication with a central broker or any other third party on each payment. An amount earned by providing resources for a remote device can be re-used for consuming remote resources from other devices. Multiple-spending of coins is persecuted by posterior fraud detection when coins are cashed by an off-line broker.

We illustrate the main benefits of our incentive system—energy-efficiency and resilience—in in-depth performance studies. By prototype-based measurements we show that using our system increases the energy consumption only marginally compared to a resource-sharing system that does not provide incentives. Furthermore, in a simulation study considering different fractions of malicious users, we show that fraud can be detected with high probability. Even in a worst case scenario with 75% of all participants being colluding malicious users only 3.2% of them get away without being punished for their fraud for a short period of time. These results clearly illustrate that the proposed system can successfully recompense energy for servicing remote resource requests with low energy-consumption by itself and high resilience to fraudulent users.

The remainder of this paper is organized as follows. Section 2 reviews existing incentive systems with focus on the design goals energy efficiency and resilience. The concepts of our incentive system are described in Section 3. Section 4 presents the mechanisms that are applied for fraud detection when cashing coins at the off-line broker. Evaluation results with respect to the energy consumption and the probability of fraud detection are provided in Section 5. Finally, concluding remarks are given.

2 Requirements and Related Work

Subsequently we present a reference scenario to show the requirements that we hold necessary to be fulfilled: Imagine a group of tourists gathering around a point of interest. Assuming that their mobile devices advertise the resources they can provide in a MANET a camera may add location tags to taken pictures by obtaining GPS readings from a nearby navigation device. Beyond offering remote access to a single resource, a device can offer remote services that combine multiple resources. For example, exploiting its GPS and WAN links a PDA can offer a service for location-tagging and gallery-upload of a picture.

Providing such services is expensive in terms of energy. That is why there must be some kind of incentive system that recompenses users who provide services to other users. Based on the described scenario we see the following requirements for such an incentive system:

- *No need of trusted hardware:* Incentive systems that depend on trusted hardware modules are applicable only on a subset of the currently widespread mobile devices. The more requirements an incentive system makes with respect to the using devices the smaller the group of potential users will become.

- *No need of always available central infrastructure:* Long-living connections to central infrastructure components via wireless WAN technology considerably decrease the battery lifetime of mobile devices [14]. The incentive system itself should not be a major consumer of energy itself.
- *No involvement of third party:* Including a third party in every payment increases the communication overhead considerably. Either an internet connection is needed every time a user wants to consume or provide a service or the involved party must be nearby in the very same MANET. Both assumptions are to strong restrictions for our use case.
- *Vendor-independent currency:* Currency that can be used only to pay a specific vendor restricts the usability in scenarios in which every participant is a potential resource provider *and* consumer.
- *Reuse of currency possible:* Currency that can be earned and spent several times allows participants to achieve liquidity by providing services. So there is a real incentive to provide resources and services.
- *No need of public key cryptography:* Creation and verification of digital signatures as well as de- and encryption of data using public key cryptography is a demanding task with respect to CPU capacity.
- *Fraud detection possible:* In case malicious users commit fraud it is a desirable property of an incentive system, that the fraud can be detected and the initiator can be tracked down.

There is plenty of related work on accounting and creating incentives in decentralized networks. According to the classification presented by Obreiter and Nimis [12] incentive schemes can be categorized into trade based and trust based systems. Examples and discussion of trust based systems can be found in [16], [6]. As good reputation only gives good return in settings with stable cooperation patterns [12] and our use case assumes flexible and often changing cooperation patterns our approach is a trade based incentive system. Subsequently we will explain in which aspects our approach differs from other trade based incentive systems.

Buttyán and Hubaux [7] present an incentive system that stimulates cooperation in MANETs by introducing a so called *nuglet counter*. This counter is incrementend whenever foreign data is forwarded and decremented whenever own data is sent. As a positive counter is required to send own data, there is a strong incentive to behave cooperatively. To avoid users to manipulate this counter is is secured by a tamper resistant security co-processor. Although this approach might be used for other resources than forwarding capacity as well, the need for a trusted hardware module on each participating device contradicts our first requirement.

Liebau et al. [19] propose a token-based accounting system for P2P-systems. Although they provide a concept for accounting without central infrastructure their scenario differs in an important point: As we focus on resource sharing on *mobile* devices it is most important to us not to spend to much energy on the incentive system itself. Otherwise the accounting and incentive system would not encourage people to share resources at all. As we want to build an incentive system that is tailored especially to the needs of battery driven mobile devices, we try to go without the means of public key cryptography. According to Rivest et al. [24] the usage of public key cryptography for securing electronic currency is rather expensive in terms of CPU load and thus also in terms of energy: They estimate that the computing time needed for verification of an RSA signature is 100 times higher than computing a hash function, generating an RSA signature takes even 10000 times longer than computing a hash function.

Other payment schemes that apply public key cryptography are P-Pay [26], Sprite [27], and the work by Blaze [4] and Abdelkader [1]. Another important difference is the value of a token. The tokens proposed by Liebau et al. do not have any intrinsic value whereas each coin that we use in our approach represents a fixed value. How this specific value can be determined is discussed in Section 3.2.

There are incentive systems that do not make heavy use of public key cryptography. However, some of them do not offer a currency that can be used for several subsequent payment operations. Millicent [20], NetPay [9], and Micromint [24] for example assume that the means of payment are redeemed after a single payment. In our use case this would mean that each participant must either possess a larger amount of electronic currency to maintain liquidity for a sufficient period of time or that he must get small amounts of electronic currency more frequently.

Bocek et al. propose a private and shared history-based incentive mechanism (PSH). However, their approach is based upon an assumption that is fundamentally different from ours: Bocek et al. assume that resource interests in the resource sharing system are asymmetric, i. e. that an entity that provides a resource to another entity is not interested in the resources that this one is sharing. In contrast we do not take that assumption but hope that resource consumers also offer resources that are attractive to resource providers. Besides, PSH also makes use of public key cryptography, which we do not for reasons that are explained above.

In PeerMint [13] a decentralized accounting system for P2P applications is provided. In contrast to our approach it tries to accomplish a system without any central component whereas we rely on a central broker although this component of our system may be reachable only casually. In order to guarantee high scalability and robustness it is built upon a structured P2P overlay network. As we are considering use cases with only a small number of users we do not base our approach on such a system that comes along with a considerable communication overhead.

Another distinguishing feature is the scale of the incentive system. As our reference scenario that was described before shows we are considering resource sharing in small MANETs. This means that the cost for implementing large scale solutions as proposed for example in [17] and [6] is not adequate.

3 An Incentive System for Resource Sharing in MANETs

Our resilient and energy-saving incentive system for resource sharing in MANETs uses electronic currency to compensate resource providers for their efforts. We use the MicroMint system [24] to create such "digital coins". A coin consists of a certain number of values that create a hash collision when used as input for a certain hash function. It is very hard in terms of computing time (and thus energy) to create such collisions but easy to verify if certain values create such collisions. This means it is hard to create coins (even on strong computing machines), but easy to check, whether a coin is valid (even on poorly equipped mobile devices). There are a couple of techniques to vary the effort that is needed to create valid coins. That way it is possible to make it unprofitable if not impossible to forge a substantial amount of coins. The coins are generated on a central system, that has plenty of resources compared to the mobile devices that are supposed to use the coins.

Coins that are constructed following the MicroMint approach are actually not made for multiple payments. They are valid only for a specified amount of time and they are supposed to be used only once. However, we see the option of using earned coins for consuming services as a major motivation to provide services. Furthermore, redeeming the coins after using them only one time would increase communication with the broker and would therefore mean higher energy expenses. Thus, we developed a system that allows to use MicroMint based coins in multiple payments. As this opens up several opportunities to commit fraud for malicious users we also present a technique to track down fraudulent users. Thus, we explore the design space that is spanned by the trade-off security vs. energy consumption.

In the remainder of this section we will clarify the assumptions that our work is built upon before we give an overview on the entire system.

3.1 Assumptions

Our incentive system and the technique for tracking attempted and committed fraud relies on several assumptions that are reasonable with respect to the use case presented in Section 1.

- Participants can be identified: Tracking down and punishing malicious users requires a way to identify each user. In a lot of incentive systems this is achieved by digital signatures that employ public key cryptography. As explained before, it is one of our explicit goals not to use such techniques because of their high demand of computing capacity and energy. Still we assume, that every participating device can be identified. This can be done for example using pre-distribution of random keys like proposed e. g. by Ramkumar et al. [21].
- There is a central, trusted component: This component that is called broker subsequently does not need a continuing connection to the resource providers and consumers, but at least once in a validity period of the used coins. The broker is not involved in any payment operation between a service provider and a service consumer. It only delivers new coins at the beginning of a validity period and redeems coins at the end of a period.
- There is a common schedule: All participants that want to take part in the fraud detection system contact the broker within a certain time frame. This could be the first day of each month for example. Participants that do not meet this deadline still can redeem their coins, however they must accept not be recompensated if the coin has been redeemed by a malicious user before.
- There are means to invoice the transactions: Just in case that the virtual currency is supposed to be bound to a real currency (which is not necessarily the case) we further assume that the broker is able to bill and withdraw the according amount of money from the participants account. Just as it is done for example by a cell phone provider. The actual arrangement of this business process is out of scope for this paper.
- Attacker model: Furthermore, we assume that malicious users cannot forge coins that pass the validity checks run by each device before accepting a coin. This assumption is reasonable as Rivest et al. [24] provide techniques that make the necessary efforts for forging coins many times higher than the costs that the legitimate issuer of the coins has to bear. Instead we consider malicious users that try to spend original coins several times. We examine both cases, a single fraudulent user and colluding fraudulent users. A single fraudulent user can try to conceal the fraud by forging transaction logs whereas a group of colluding fraudulent users can even come to an agreement to incriminate a third party of having committed the fraud. We assume such colluding fraudulent users to act in fixed groups. We think this is reasonable as the most likely appearance of such a colluding group is a person that owns serveral mobile devices that she configured to commit fraud.

3.2 Overview

The architecture of our system consists out of three major components: broker, service consumer and service provider. Mobile devices might either act as both service consumer *and* service provider or incorporate only one of the two functions. At least once in the validity period of a coin (that might be a month, e. g.) service providers and consumers need a connection to the broker. This can be done either through a wireless WAN interface or when the mobile devices eventually gain access to the internet via WiFi-LANs. Then the mobile devices can get new valid coins or redeem the ones that they earned for providing services. Note that this requires only a sporadic connection between broker and participating devices. Each coin represents a fixed, small monetary value. This could be 0.1 cent, for example.

Every time a mobile device wants to offer a service, an adequate fee for using the service is also published within the service advertisement. As we want to recompense service providers for their energy expenses, the price to pay for using a service should be related to its energy cost. However, the value of energy for the user of a mobile device varies depending on usage

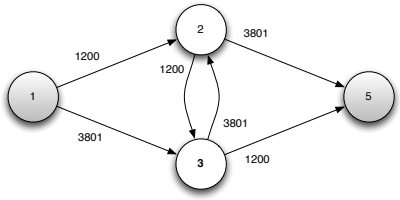

Figure 1 Transactions between honest users. Node one and five standing for the broker's dispensing and redeeming part.

scheme, remaining battery charge and chances to recharge the battery. There are several approaches in related work how the price of a service can be adjusted with regard to those variable conditions, e. g. in [3].

As the price of a service is published within the service advertisement, a service consumer can base the choice of provider on the announced service cost. On requesting the service, the service consumer transfers the specified number of coins to the service provider. If the service provider fulfills its obligation the service consumer marks the corresponding coins as spent. If the service provider does not meet its duty the service consumer adds the provider's id to a blacklist. In case of a successful payment both participants create transaction logs. A transaction log contains the unique id of all coins involved in the payment and the ids of service provider and consumer. These transaction logs can be used for the posterior fraud detection presented subsequently in Section 4.

4 Posterior Fraud Detection

According to the attacker model presented in Section 3.1 only multiple-spending of coins will discussed in this work.

The only chance of detecting the reuse of already spent coins at user-level is, if the coin already exists in the local coin set and a user wants to pay this user with the same coin. In all other cases, fraud can only be detected by the broker after all coins have been redeemed. Therefore every client saves a transaction log for received and spent coins including the involved user-IDs. The transaction log is handed over to the broker when redeeming the coins. If the broker detects that one or more coins are redeemed by more than one user, he uses a graph built from the collected transaction logs. This graph is used for detecting the fraudulent users. The nodes thereby represent the users, the edges the transfer of coins (identified by their coin-IDs) between these users (as shown in Figure 1, e. g.).

Honest users, redeeming all received coins at the end of the validity period, should have the same amount of incoming to outgoing edges, as shown in Figure 1. In this case we call all edges verified as for all of them two congruent transaction log entries can be found. Only fraudulent users or users trading with fraudulent users will show an imbalance or have unverified transactions, where only one log entry for the payment can be found.

As a fraudulent user is willing to hide his fraud, he will likely manipulate his own transaction log by either keeping only one of the entries for the multiple-spent coin or adding bogus entries telling he received the coin multiple times. This manipulation leads to notcongruent transaction log entries as shown in Figure 2. We call these unverified edges.

The broker scans the transaction logs for these unverified transactions and marks the involved users. A naïve approach would be to only mark the users with outgoing unverified edges as cheaters. Then two or more users could conspire to draw the broker's attention on another user as shown in Figure 3. In this example user 3 could hand over a coin received from user 2 to node 4. Both colluding fraudulent users could delete the transaction log for this transaction and create transaction logs that indicate that they received the coin from user 2.

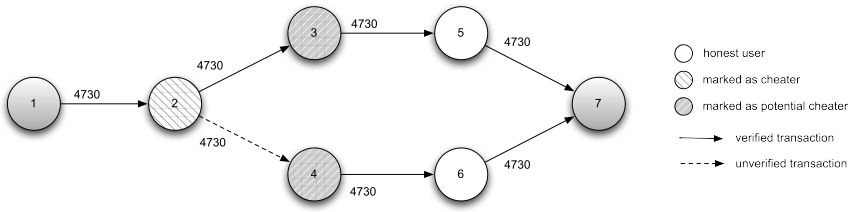

Figure 2 Transactions including a cheaters (node 2) double spending of a coin. Node one and seven depict the broker that is dispensing and redeeming the coins.

Algorithm 1 explains the tracking algorithm that is used to face the challenge of colluding fraudulent users. If an unverified edge is detected, the user with the outgoing unverified edge *and* all users receiving the involved coin from this user are marked as possibly fraudulent. The user with the outgoing edge hereby gets a score of two, the users with the incoming edge receive a score of one (line 5 and line 10). Considering the example from Figure 3 user 2 gets a score of 2, users 3 and 4 get a score of one each. These scores were chosen with respect to the thresholds used in Algorithm 2. A user must occur at least two times spending a duplicated coin or three times receiving a duplicated coin before he is considered a malicious user.

Thereafter the set of the receiving users (in our example this is {3,4}) is put into a list of potential conspiratorial groups. If the group has been added before, its group score is increased (lines 14 - 19). The score of the users and the group score is used to determine whether a user is treated as fraudulent user or as victim of a conspiracy. How this is done is shown in detail in Algorithm 2. A fraudulent user who spends a coin several times without forging transaction logs will be caught as he has more verified outgoing transactions than incoming transactions for the very same coin (lines 22 - 27). He gets a score of 3 (line 25) which is higher than the COIN-FRAUD-THRESHOLD (=2).

Algorithm 1 Algorithm to detect fraud

```
 1: for all coins in multipleRedeemedCoins as coin do
 2:     for all unverified transactions of coin as ut do
 3:         source = ut.source
 4:         if source has more than 1 outgoing transactions then
 5:             setScore(source, coin.id, 2)
 6:             possibleConspirators = new List
 7:             for all outgoing transactions of source as ot do
 8:                 receiver = ot.receiver
 9:                 if receiver has at least one outgoing transaction && getScore(receiver) == 0 then
10:                     setScore(receiver, coin.id, 1)
11:                     possibleConspirators.add(receiver)
12:                 end if
13:             end for
14:             if suspectedGroups.contains(possibleConspirators)) then
15:                 incrementGroupscore(possibleConspirators)
16:             else
17:                 suspectedGroups.add(possibleConspirators)
18:                 setGroupscore(possibleConspirators, 1)
19:             end if
20:         end if
21:     end for
22:     for all verified transactions of coin as vt do
23:         source = vt.source
24:         if source has more outgoing than incoming transactions for coin then
25:             setScore(source, coin.id, 3)
26:         end if
27:     end for
28: end for
```

The process of actually resolving the fraud is described in Algorithm 2: The marking of multiple users as possible fraudulent requires the adoption of thresholds for the scores to

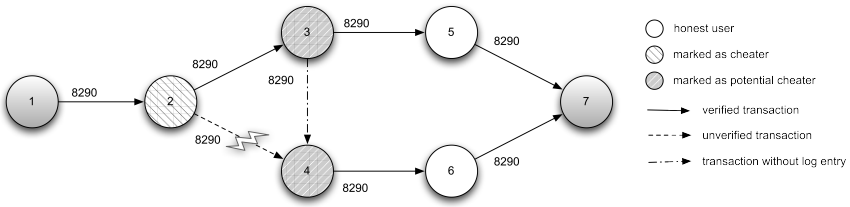

Figure 3 Colluding cheaters (node 3 and 4) blame an honest user for the double spending by forging history entries.

avoid bringing innocent users to justice. First, only if the score for one coin is above the COIN-FRAUD-THRESHOLD (=2) for the user, he is regarded when reallocating the costs of the excessive redemption (line 8). This means a user must be involved at least two times into suspicious transactions before he is charged for a fraud. Users with a low score are likely to be innocent while the probability of a fraudulent user is higher the more he is being involved in unverified transactions for the coin.

Nevertheless a user whose score for all the coins is below the threshold could still be fraudulent by just double spending each coin and hereby keeping a low score for each coin. So the total score over all unverified transactions also has to be below a second threshold, TOTAL-FRAUD-THRESHOLD (=4), before the user is marked as innocent (line 8). The values for both thresholds provided good results, however, future work might show that there are better settings. Generally higher values for the thresholds bear the risk of an increased number of false negatives whereas lower thresholds increase the number of false positives.

After the broker's detection of who of the users is fraudulent, he uses the scores of these users to reallocate the values of the excessive redemption and debits their user accounts. The amount a convicted user is held liable for depends on his score. So users that are more likely to be cheaters have to pay more.

The broker maintains a list where he keeps a record of all frauds committed by the users. When a user reaches a predefined limit he will be blocked out of the system. The blocked users are regularly distributed to all other users, so a blocked user can not trade anymore with the broker or any other users.

Note that Algorithm 1 can be easily parallized. This is important as checking the transaction logs might become a bottleneck of the system. Future work will show whether and how it is possible, to parallize or speed up Algorithm 2.

5 Evaluation

The evaluation of our approach was twofold: First, we examined how much energy is spent for the actual payment. Related work states clearly, that systems based on hash functions are cheaper than for example RSA signatures. Rivest et al. explain that hash functions are about 100 times faster than RSA signature verification [24]. Still, it remains to show that the energy cost of the incentive system is small compared to the resource sharing framework itself. For this reason we did measurements to determine the additional energy overhead caused by our incentive system.

Second, we analyzed how good the mechanism to detect fraudulent users actually works. Both are critical issues with respect to user acceptance. The energy measurements were done using a prototype whereas we did a simulative evaluation of the fraud detection mechanism.

5.1 Energy Consumption

The prototype [11] for measuring the energy overhead that is caused by the incentive system consists out of two Nokia N810 devices. We implemented a resource sharing framework based on OSGi [2, 23] and R-OSGi [22]. Resources and services that are supposed to be shared

Algorithm 2 Algorithm to resolve fraud

```
1: //remove possibleConspirators from suspectedGroups if groupscore is below threshold
2: cleanSuspectedGroups();
3: for all coins in multipleRedeemedCoins as coin do
4:    conspiracyDetected = false
5:    totalScore = 0
6:    fraudulentUsers = list of users with score > 0 for this coin
7:    for all users in fraudulentUsers as user do
8:       if getScore(user, coin.id) > COIN-FRAUD-THRESHOLD || getScoreForAllCoins(user) >
      TOTAL-FRAUD-THRESHOLD then
9:          score = getScore(user, coin.id)
10:         if user is in suspectedGroups then
11:            score = (number of possibleConspirators groups the user appears in for coin) * 3
12:            conspiracyDetected = true
13:         else
14:            if conspiracyDetected && score < COIN-FRAUD-THRESHOLD then
15:               fraudulentUsers.remove(user) // Delete innocent user when conspiracy is detected
16:               score = 0
17:            end if
18:         end if
19:         totalScore += score;
20:         //User is below thresholds
21:      else
22:         fraudulentUsers.remove(user) // Delete fraudulent users below threshold
23:      end if
24:   end for
25:   doubleRedeems = number how often this coin has been redeemed
26:   ratio = doubleRedeems / totalScore
27:   for all fraudulent users of coin as user do
28:      valueToPay = round((getScore(user, coin.id) * ratio) + accRoundingError);
29:      accRoundingError = ((getScore(user, coin.id) * ratio) + accRoundingError) - valueToPay;
30:   end for
31: end for
```

as well as the broker and the software for doing the actual payments are implemented as remotely accessible OSGi bundles. One of the devices runs a simple service that changes the characters of a word from upper- to lowercase and vice versa. The service is advertised in a wireless ad hoc network. The other N810 device acts as service consumer. It requests the service and sends the needed number of digital coins to the service provider. The service provider checks the validity of the payment and if there is no reason to complain it provides the service. We used an SNMD [15] to measure the current drawn from the battery during the process of providing the service and checking the validity of the coins. In each run of the experiment the service was requested, provided and payed for 100 times. We conducted 10 runs of this experiment. The measured energy cost does not include the cost for redeeming the coins. However, this occurs rather infrequently (e.g. once a month) compared to the actual payment operations and thus it can be neglected.

As a reference point we repeated the entire experiment without doing any payment. Figure 4 shows the electric charge that was drawn from the battery in these two experiments. In average, checking the payment for valid coins adds an overhead in terms of energy of about 3%. This is mainly due to a slightly longer runtime (in average 4.54%). Note, that the relative overhead will become much smaller for services that consume more energy than our modest dummy service. So the 3% overhead can be seen as an upper bound for the overhead that is caused by our incentive scheme. The fact that there are even some runs with payment that consume less energy than some of the runs without payment shows, that the overhead of the payment system in fact is very small. Obviously it can be outweighed by other factors that are not related to the incentive system, e. g.background processes scheduled by the operating system or necessary retransmissions due to a busy WiFi channel.

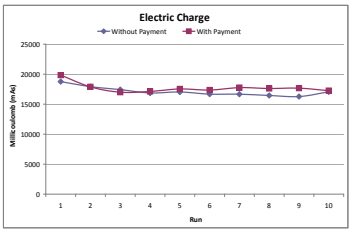

Figure 4 Comparison of energy consumption with and without the incentive system

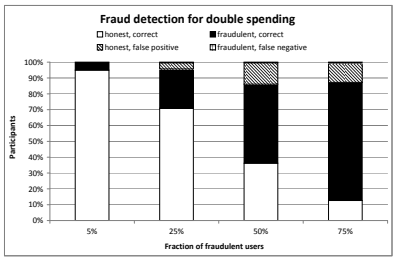

Figure 5 Detection rate of single fraudulent users.

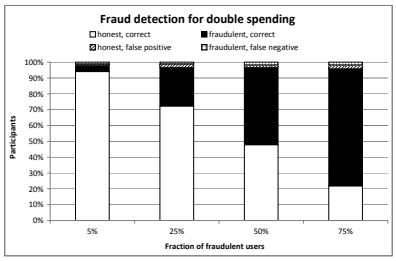

Figure 6 Detection rate of conspiratorial fraudulent users.

5.2 Fraud Detection

The effectiveness of our technique for posterior fraud detection was analyzed in simulations. The simulations were done with an event based simulator written in Java. In each run 100 participants did 1000 payments in total with a randomly chosen transaction partner. The fraction of fraudulent users was set to 5, 25, 50 or 75 percent. In each configuration we did 10 runs with different random seeds. We did the entire experiment twice, once for the attacker model of a single malicious user that tries to use his coins more than once, and a second time for a more powerful attacker that is able to cooperate with other malicious users. Such collusive participants can try to spend coins multiple times and both blame a third party, that is actually innocent.

After each run the broker did its graph based fraud detection using the transaction logs provided by the service providers and consumers. The results of the fraud detection process were compared to the actual behavior of the users and each decision that was taken by the broker was classified to be one of the following:

- honest, correct: An honest participant was correctly detected as an honest user.
- fraudulent, correct: A fraudulent participant was correctly detected as a fraudulent user.
- honest, false positive: An honest participant was wrongly detected as a fraudulent user.
- fraudulent, false negative: A fraudulent participant was not detected.

Figure 5 shows the results in case that fraudulent users act on their own and do not collude. Our key finding from this result is that the number of malicious participants that are not detected by the broker is close to zero (0.6 % in average). As malicious users hardly have any chance to get away without being punished there is only little appeal to behave badly.

Figure 6 shows even better results for groups of collusive cheaters. Fraudulent users colluded in teams of two, spent coins multiple times and blamed a third, innocent user for it.

They are detected, because they appear several times as a suspicious group, according to the assumption presented in Section 3.1. The broker and its detection mechanism work pretty well. Less than 5 % of the users are classified wrong, even in the worst case scenario with 75% of fraudulent users. A substantial amount of wrong classifications is due to colluding malicious users that did not spend a coin twice but were involved in blaming an innocent user. Such users are often detected as cheaters and this decision is considered to be a false positive, here. If this was not the case the rate of wrong detections would even be lower (2.4 % of all users in the case of 75 % fraudulent users). Relating to the number of fraudulent users this means that 3.2 % of the cheaters get away.

6 Conclusion and Outlook

In this paper we argued that resource sharing systems for MANETs lack an incentive system tailored to this specific use case. Our approach provides the means to recompense service providers for the energy they spent on service provision. As neither public key cryptography nor communication with a third party is needed on each payment we could eliminate two major drivers of energy costs. This results in only very little additional energy consumption ($< 3\,\%$) caused by our incentive system as shown by prototype measurements. Although the abandonment of powerful cryptographic tools and the reuse of currency create opportunities for abuse we showed through simulations that almost all malicious users can be tracked down by our system for posterior fraud detection by an off-line broker. In a worst case scenario with 75 % of the users being cheaters only 3.2 % of them get away without being caught. This proves that it is possible to build an incentive system that suites the needs of resource sharing in MANETs: being cheap with respect to its energy consumption and still effective on preventing abuse.

In the near future we plan to compare the energy consumption of our incentive system to solutions that apply digital signatures for payment transactions. Further we want to investigate on how to further decrease the number of false positives in our fraud detection system.

References

1 M. Abdelkader, N. Boudriga, and M. S. Obaidat. Secure Grid-Based Multi-Party Micropayment System in 4G Networks. In *ICE-B*, pages 137–148, 2007.

2 OSGi Alliance. *OSGi Service Platform, Release 3*. IOS Press, Inc., Fairfax, 2003.

3 G. Ananthanarayanan, V. Padmanabhan, L. Ravindranath, and C. Thekkath. Combine: Leveraging the Power of Wireless Peers through Collaborative Downloading. In *Proc. 5th International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM New York, 2007.

4 M. Blaze, J. Ioannidis, and A. D. Keromytis. Offline Micropayments without Trusted Hardware. In *Proc. 5th International Conference on Financial Cryptography*, pages 21–40. Springer-Verlag, 2002.

5 R. Bless, C. Hübsch, C. Mayer, and O. Waldhorst. SpoVNet: An Architecture for Easy Creation and Deployment of Service Overlays. In Anand R. Prasad, John F. Buford, and Vijay K. Gurbani, editors, *Advances in Next Generation Services and Service Architectures*. River Publishers, 2011. *To appear*.

6 T. Bocek, M. Shann, D. Hausheer, and B. Stiller. Game theoretical analysis of incentives for large-scale, fully decentralized collaboration networks. In *IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*. , pages 1–8, 2008.

7 L. Buttyán and J.-P. Hubaux. Stimulating Cooperation in Self-Organizing Mobile Ad hoc Networks. *Mobile Networks and Applications*, 8(5):579–592, 2003.

8 A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G.-S. Ahn. The Rise of People-Centric Sensing. *IEEE Internet Computing*, 12:12–21, 2008.

9 X. Dai and J. Grundy. NetPay: An Off-line, Decentralized Micro-Payment System for Thin-Client Applications. *Electronic Commerce Research and Applications*, 6(1):91–101, 2007.

10 J. Furthmüller and O.P. Waldhorst. A Survey on Grid Computing on Mobile Consumer Devices. In N. Antonopoulos, G. Exarchakos, M. Li, and A. Liotta, editors, *Handbook of Research on P2P and Grid Systems for Service-Oriented Computing*, pages 313–363. IGI-Global, 2010.

11 J. Furthmüller, S. Becker, and O.P. Waldhorst. An Energy Manager for a Mobile Grid. Demo on MobiSys09, Krakow, Poland, 2009.

12 *A Taxonomy of Incentive Patterns - The Design Space of Incentives for Cooperation.* In M.P. Singh G. Moro, C. Sartori, editors, *Agents and Peer-to-Peer Computing* volume 2872 of *Lecture Notes in Computer Science*, pages 89–100. Springer Berlin / Heidelberg, 2003.

13 D. Hausheer and B. Stiller. PeerMint: Decentralized and Secure Accounting for Peer-to-Peer Applications. In R. Boutaba, K. Almeroth, R. Puigjaner, S. Shen, and J.P. Black, editors, *NETWORKING 2005*, volume 3462 of *Lecture Notes in Computer Science*, pages 40–52. Springer Berlin / Heidelberg, 2005.

14 H. Haverinen, J. Siren, and P. Eronen. Energy Consumption of Always-On Applications in WCDMA Networks. In J. Siren, editor, *Proc. IEEE 65th Spring Vehicular Technology Conference (VTC)*, pages 964–968, Dublin, Ireland. IEEE Vehicular Technology Society, 2007

15 A. Hergenröder, J. Wilke, and D. Meier. Distributed Energy Measurements in WSN Testbeds with a Sensor Node Management Device (SNMD). In M. Beigl and F.J. Cazorla-Almeida, editors, *Workshop Proceedings of the 23th International Conference on Architecture of Computing Systems*, pages 341–438, Hannover, Germany. VDE Verlag, 2010.

16 S. Kaune, K. Pussep, G. Tyson, A. Mauthe, and R. Steinmetz. Cooperation in P2P Systems through Sociological Incentive Patterns. In K. Hummel and J. Sterbenz, editors, *Self-Organizing Systems*, volume 5343 of *Lecture Notes in Computer Science*, pages 10–22. Springer Berlin / Heidelberg, 2008.

17 S. Kotrotsos, P. Racz, C. Morariu, K. Iskioupi, D. Hausheer, and B. Stiller. Business Models, Accounting and Billing Concepts in Grid-Aware Networks. In O. Akan, P. Bellavista, J. Cao, F. Dressler, D. Ferrari, M. Gerla, H. Kobayashi, S. Palazzo, S. Sahni, X.(S.) Shen, M. Stan, J. Xiaohua, A. Zomaya, G. Coulson, A. Doulamis, J. Mambretti, I. Tomkos, and T. Varvarigou, editors, *Networks for Grid Applications*, volume 25 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 27–34. Springer Berlin Heidelberg, 2010.

18 K. Lahiri, A. Raghunathan, S. Dey, and D. Panigrahi. Battery-driven system design: a new frontier in low power design. In A. Raghunathan, editor, *Proc. 7th Asia and South Pacific and the 15th Int. Conference on VLSI Design Design Automation Conference Proceedings ASP-DAC 2002*, pages 261–267, Bangalore, India. Center for Embedded Computer Systems, 2002.

19 N. Liebau, V. Darlagiannis, A. Mauthe, and R. Steinmetz. Token-Based Accounting for P2P-Systems. In W. Brauer, P. Müller, R. Gotzhein, and J. Schmitt, editors, *Kommunikation in Verteilten Systemen (KiVS)*, Informatik aktuell, pages 16–28. Springer Berlin Heidelberg, 2005.

20 M.S. Manasse. The Millicent Protocols for Electronic Commerce. In *Proceedings of the 1st conference on USENIX Workshop on Electronic Commerce - Volume 1*, pages 9–9. USENIX Association, New York, 1995.

21 M. Ramkumar and N. Memon. An efficient random key pre-distribution scheme. In *Global Telecommunications Conference (GLOBECOM). IEEE*, volume 4, pages 2218 – 2223 Vol.4, 29. 2004.

22 J. Rellermeyer, G. Alonso, and T. Roscoe. R-OSGi: Distributed Applications Through Software Modularization. In *Proc. ACM/IFIP/USENIX 8th International Middleware Conference*, pages 1–20. 2007.

23 J. Rellermeyer and G. Alonso. Concierge: A Service Platform for Resource-Constrained Devices. In *Proc. EuroSys 2007 Conference*, Lisbon, Portugal. ACM, 2007.

24 A. Shamir R.L. Rivest. *Security Protocols*, chapter PayWord and MicroMint: Two simple micropayment schemes, pages 69–87. Springer Berlin/Heidelberg, 1997.

25 D. Saha and A. Mukherjee. Pervasive Computing: A Paradigm for the 21st Century. *IEEE Computer*, 36(3):25–31, 2003.

26 B. Yang and H. Garcia-Molina. PPay: Micropayments for Peer-to-Peer Systems. In *Proceedings of the 10th ACM conference on Computer and communications security*, pages 300–310, Washington D.C., USA. ACM, 2003.

27 S. Zhong, J. Chen, and R. Yang. Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks. *Proceedings of IEEE INFOCOM*, pages 1987—1997, 2002.