MobiCent: a Credit-Based Incentive System for Disruption Tolerant Network

Bin Bin Chen
Department of Computer Science, National University of Singapore
Email: chenbinb@comp.nus.edu.sg

Mun Choon Chan
Department of Computer Science, National University of Singapore
Email: chanmc@comp.nus.edu.sg

Abstract

When Disruption Tolerant Network (DTN) is used in commercial environments, incentive mechanism should be employed to encourage cooperation among selfish mobile users. Key challenges in the design of an incentive scheme for DTN are that disconnections among nodes are the norm rather than exception and network topology is time varying. Thus, it is difficult to detect selfish actions that can be launched by mobile users or to pre-determine the routing path to be used.

In this paper, we propose *MobiCent*, a credit-based incentive system for DTN. While MobiCent allows the underlying routing protocol to discover the most efficient paths, it is also incentive compatible. Therefore, using MobiCent, rational nodes will not purposely waste transfer opportunity or cheat by creating non-existing contacts to increase their rewards. MobiCent also provides different payment mechanisms to cater to client that wants to minimize either payment or data delivery delay.

I. Introduction

Disruption Tolerant Networks (DTN) are characterized by intermittent connectivity. Such networks are assumed to experience frequent, long duration partitioning and often lack an end-to-end contemporaneous path [1]. DTN is initially proposed for environments such as inter-planetary networks and disaster relief team networks. Recently, it has also been applied to other environments such as social networks and vehicular networks.

In DTN routing, as contacts are often unpredictable, forwarding (and replication) of data among mobile relays happens in an opportunistic manner. In order to increase delivery success ratio and reduce delivery delay, many multi-copy DTN routing protocols [2] [3] [4] [5] [6] have been proposed. In these protocols, multiple copies of the data simultaneously propagate in the network along different paths. Thus, different from routing in mostly-connected mobile ad-hoc networks, the feasible delivery paths in DTN are revealed only when copies of the data reach the destination.

When mobile nodes are managed by autonomous and selfish parties, an incentive scheme is needed to foster cooperation among participants in the DTN. There are two key challenges in designing the scheme. First, disconnections among nodes are the norm rather than exception. As a result, selfish actions are extremely difficult to detect. Second, as contacts are unpredictable, the delivery paths cannot be predetermined, but must be discovered along with the forwarding of data instead.

In this paper, we present *MobiCent*, a credit-based system to support Internet access service in a heterogeneous wireless network environment. In this environment, a mobile device is capable of operating in two modes. It can use a long-range low-bandwidth radio (e.g., cellular interface) to maintain an always-on connection while using a short-range and high-bandwidth link (e.g., Wi-Fi) to opportunistically exchange large amount of data with peers in its vicinity. The short range links tend to be intermittent because of node mobility. Thus, the exploitation of these intermittent contacts requires the use of a DTN approach. In our earlier work [6], we demonstrate the benefit of employing DTN routing to improve Internet access performance for vehicles. This approach can also be used to enhance performance of mobile social networks (e.g. [**?**], [7]), where people communicate using low power wireless mobile devices.

We make the following contributions in this paper:

1) We identify *edge insertion attack* and *edge hiding attack* as the two major forms of attacks in a DTN environment. It is extremely difficult to detect them in DTN, and they can seriously degrade the performance of DTN routing.

2) We take algorithmic mechanism design approach [8] to address the two attacks, and identify the necessary conditions under edge insertion attack for a payment scheme to be incentive compatible, i.e., truthful participation is adopted by selfish nodes.

3) We propose incentive-compatible payment mechanisms to cater to client that wants to minimize either payment or data delivery delay.

MobiCent does not require detection of selfish actions as it provides incentives for selfish nodes to behave honestly. In addition, MobiCent does not require pre-determined routing path. It works on top of existing DTN routing protocols to ensure that selfish actions do not result in larger rewards. To the best of our knowledge, MobiCent is the first incentive compatible scheme proposed for replication based DTN routing protocols.

The rest of the paper is organized as follows. In Section II, we describe related incentive schemes and give an overview of the algorithmic mechanism design approach. Section III presents the system model and formulates the attack model and the path revelation game. The message exchange protocol to support MobiCent is presented in Section IV. We analyze the payment scheme required to thwart edge insertion attack in Section V, followed by the mechanisms designed to combat edge hiding attack in Section VI. Evaluation is presented in Section VII. We conclude in Section VIII.

II. Related Work

A. Incentive Schemes in Wireless Networks

It is widely agreed that some form of incentive is needed for wireless networks with user-contributed forwarding (e.g. mobile ad hoc networks) to overcome the free-riding problem, i.e., requesting others for forward his packets, but avoiding to transmit others' packets. The three main incentive mechanisms being studied in literature are reputation, barter (or Tit-for-Tat), and virtual currency.

In general, a reputation scheme is coupled with a service differentiation scheme. Contributing users possess good reputations and receive good service from other peers. For example, users in [**?**] build up their reputation scores by forwarding packets for others, and are rewarded with higher priority when transferring their own packets.

Reputation-based approach is known to suffer from sybil attack [9] and whitewashing attack [10]. [9] coins the name of sybil attack. In a sybil attack, a single malicious peer generates multiple identities that collude with one another. Multiple colluding peers may boost one another's reputation scores by giving false praise, or punish a target peer by giving false accusations. In a whitewashing attack [10], a peer defects in every transaction, but repeatedly leaves and rejoins the system using newly created identities, so that it will never suffer the negative consequences of a bad reputation. The availability of cheap pseudonyms in our target environment makes reputation systems vulnerable to such attacks.

A recent work [11] proposes the use of pair-wise Tit-for-Tat (TFT) as incentive mechanism for DTNs. They enhance their TFT mechanism with generosity and contrition to address the bootstrapping and link variation problem. Tit-for-Tat does not suit our target environment, because in such environments, one peer is likely to want much more service from another peer than it could provide to that peer. In such asymmetric settings, a credit-based system can better support the asymmetric transactions needed.

The use of virtual currency for incentives has also been proposed in wireless networks. The largest community-based Wi-Fi ISP FON [12], has officially used its Wi-Fi Money to encourage its member to cooperate. [13] proposes nuglets that serve as a per-hop payment in a tamper-proof security module in each node to encourage forwarding. [14] discusses a micro-payment scheme to encourage collaboration in multi-hop cellular networks and [15] proposes Sprite, a cheat-proof, credit-based system for stimulating cooperation among selfish nodes in mobile ad hoc networks. [16] [17] propose schemes based on use of Vickrey-Clarke-Groves (VCG) pricing. More discussion of VCG will be given in the following section.

These credit-based schemes cannot be directly applied in DTNs due to the following reasons. First, a common assumption adopted in these schemes is that an end-to-end connection between the source and the destination is established before the data forwarding occurs. Second, the reported schemes are mainly designed for single path forwarding. Recently, [18] proposes a secure credit based incentive scheme for DTNs. However, the emphasis of [18] is on generation and verification of secure bundle and does not deal with pricing. All existing payment schemes are vulnerable under sybil attack, as we will show in Section V.

[19] and [20] propose mechanisms to detect sybil attack in wireless networks. The basic idea is to test the resource

of a node. Based on the observation that a given node only has limited resource (say, a single Wi-Fi radio), a testing node can assign its neighbors into different channels, and randomly probes for a neighbor in the specified channel for it. If a node mimics several sybils which are assigned to different channels, as it can only appear in one channel in any given time, the probability that one of its sybils is caught is high. [14] relies on statistic techniques to detect sybil attack in multi-hop cellular networks over a long period of time. However, sybil attack is much more difficult to detect in DTN since disconnections are the norm and high user population dynamic is expected. As a result, these techniques cannot be applied.

B. Game Theory and Algorithmic Mechanism Design

Game theory aims to model situations in which multiple participants select strategies that have mutual consequences. Following the definitions used by Nisan et al. in [8], a game consists of a set of n players, $1, 2, ..., n$. Each player i has his own set of possible strategies, say S_i. To play the game, each player i selects a strategy $s_i \in S_i$. Let $s = (s_1, ...s_n)$ denote the vector of strategies selected by the players and $S = \times_i S_i$ denote the set of all possible ways in which players can pick strategies. The vector of strategies $s \in S$ selected by the players determines the outcome for each player. If by using a unique strategy, a user always gets better outcome than using other strategies, independent of the strategies played by the other players, we say that the strategy is the user's *dominant strategy*. If users select strategies such that, no player can unilaterally change its strategy to gain more payoff, we say that the game reaches a *Nash equilibrium.*

Algorithmic mechanism design [8] is a subarea of game theory which deals with the design of games. It studies optimization problems where the underlying data is *a priori unknown* to the algorithm designer, and must be implicitly or explicitly elicited from selfish participants (e.g., via a bid). The high-level goal is to design a protocol, or "mechanism", that interacts with participants so that *selfish behavior yields a desirable outcome*. In particular, when adoption of truth-telling by all participants is the unique Nash equilibrium of a game, we say the mechanism is *incentive compatible* [1]. Auction design is the most popular motivation in this area, though there are many others.

Among auction games, our work is closest to the well-studied *path auction game*. In this game, there is a network $G = (V, E)$, in which each edge $e \in E$ is owned by an agent. The true cost of e is private information and known only to the owner. Given two vertices, source s and destination t, the customer's task is to buy a path from s to t. Path auction games have been extensively studied and much of the literature has focused on the Vickrey-Clarke-Groves (VCG) mechanism. In the VCG mechanism, the customer pays each agent on the winning path an amount equal to the highest bid with which the agent would still be on the winning path. This mechanism is attractive as it is incentive compatible.

Existing works [21] [22] have shown that VCG is vulnerable to false-name manipulation, a form of sybil attacks. Furthermore, it is well known that VCG is not frugal for path auction game [23] [24] [25], i.e., a VCG based incentive compatible scheme may result in very large payment for the client.

[1]This definition is more general than the commonly-adopted definition, which requires truth-telling to be dominant strategy of all participants.

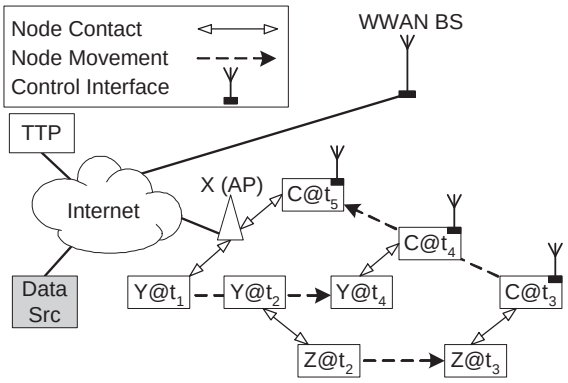


Fig. 1: MobiCent Framework

A key difference between our work and the work on path auction game is that in our work the contact graph is the information to be elicited from participants, while in the latter, topology is static and is known to all.

III. System Model and Problem Formulation

A. System Model

Based on the target future mobile communication environment, *MobiCent* assumes that nodes can have access to two different networks. All nodes are connected to a mostly disconnected network, where short range and high bandwidth links are used for data transfer. At the same time, some of the nodes (in particular the source and destination nodes) have access to a mostly available network, where long range and low bit rate links are used for control messages.

The network architecture assumed for *MobiCent* is shown in Figure 1. The components are:

- **Trusted Third Party (TTP)** stores key information for all nodes and provides verification and payment services.
- **Helpers** are mobile or static nodes (X, Y, Z in the figure) that will help in data relaying using the high speed, short range and intermittent link. These nodes do not need to have a highly available control channel.
- **Mobile Clients** are the destination nodes (C in the figure) which initiate downloading. We assume that mobile clients have high-bandwidth intermittent links for data transfer and highly available but low bit rate links for control messages.

A typical download in *MobiCent* begins with the mobile client requesting data from a data source which can be another mobile node or a data store / web server in the Internet. In the former case, the mobile source node needs access to the control channel in order to initiate packet transfer. In the latter case (as studied in our earlier work [6]), the destination node obtains the data via some access points (APs). These APs are special helpers with Internet access, and they are the data sources within the wireless domain. As an example, data for a request initiated by client C before time t_1 can be transferred from AP X to Y at time t_1, Y to Z at time t_2 and finally to C at time t_3. Due to data replication, C can also receive data from Y at t_4 and the AP at t_5. Different paths complement one another, as each of them is subject to uncertainty.

A detail description of the system including the message exchange protocol is presented in Section IV. We will first present a brief overview here. Standard cryptographic techniques and en-route onion encryption [26] are used to *prevent free riding*, *restrict strategy set of participants* and *handle dispute among relays and client*. More specifically, each relay encrypts the data payload with a one time symmetric key before forwarding it. The key is also sent along with data in encrypted form, such that only the TTP can recover the keys. Thus, after a client receives the encrypted data, the only way for the client to retrieve the decrypted data is to make payment to the TTP in exchange for the encryption key(s). Similarly, the only way for the relay to get payment is to be involved in forwarding. Note that the lightweight message exchange protocol handles a wide array of attacks, but it cannot prevent both client and relays from launching edge insertion attack and edge hiding attack, which will be described in detail in Section III-C. To address these attacks, an incentive compatible payment scheme is needed.

B. MobiCent and DTN Routing

MobiCent runs on top of a given DTN routing module, and does not rely on any specific routing protocol. We first present a generic model of DTN routing. When two nodes meet, they exchange metadata on the packets they have in their respective buffers. Based on the information exchanged, each node decides which packets it wants the other node to transfer (replicate) to it. The order of the packet transfer depends on the priority a node associates with each packet. The amount of data that can be transferred in a single contact is dependent on the duration of the opportunistic contact.

Various DTN routing protocols differ mainly on how each packet's priority is determined. In the simplest version, all packets have the same priority. However, such simple stateless epidemic routing is not efficient, and researchers have proposed many improvements. For example in PRoPHET [27], direct and indirect contact histories are used. In MaxProp [3], a combination of a few parameters, including contact history and packet hop count, are used to determine a packet's priority.

MobiCent works by setting the client's payment and the relays' rewards so that nodes will behave truthfully. Therefore, nodes will always forward packets without adding phantom links, and never waste contact opportunity unless the reward is inadequate or it is the decision of underlying routing protocol. As a result, the (best) forwarding paths that should be discovered by the given routing protocol through replication and forwarding will be discovered.

C. Path Revelation Game

Before formulating the problem as a *path revelation game*, we first define some terminologies.

Definition 1: An **edge** e represents the opportunistic contact between two nodes, through which data can be forwarded between them. Formally, an edge e is defined by the two nodes $\{v_1, v_2\}$ in contact (referred to as the edge's vertices) and the contact time $t(e)$ [2].

[2]For easy presentation, we assume contacts do not overlap and have enough capacity to exchange data. Thus, contact duration and capacity are omitted.

For example, Figure 2 plots the scenario depicted in Figure 1 as a contact graph over time axis. In the figure, X meets Y at time t_1, and the corresponding edge is denoted as $e = (\{X, Y\}, t_1)$, where X and Y are e's vertices. Given a node v, the set of edges containing it as a vertex is denoted as $E(v)$.

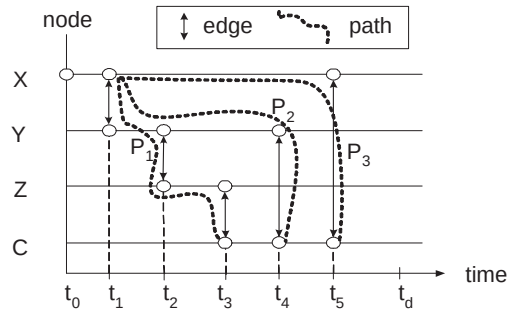


Fig. 2: A contact graph plotted over time axis

Definition 2: A **contact graph** is denoted by $G = (V, E)$, where V is the set of all nodes in the system, and E is the set of edges among the nodes.

In Figure 2, $V = \{X, Y, Z, C\}$, while $E = \{(\{X, Y\}, t_1), (\{Y, Z\}, t_2), (\{Z, C\}, t_3), (\{Y, C\}, t_4), (\{X, C\}, t_5)\}$.

Definition 3: A **forwarding path** is a sequence of nodes from source to destination, such that, from each of its nodes, there is an edge to the next node in the sequence, and edges appear in non-decreasing contact time.

Given a path P, $Relay(P)$ is the set of relays on the path. Note that source is considered as a relay. The number of relays on path $|Relay(P)|$ is defined as the length of the path. A path P with length n is called a n-hop path. At the contact time of its last edge, a path P is *revealed* to the destination.

In Figure 2, there are three paths, where P_1 consists of three edges: $(\{X, Y\}, t_1)$, $(\{Y, Z\}, t_2)$, and $(\{Z, C\}, t_3)$ in sequence; P_2 consists of two edges: $(\{X, Y\}, t_1)$ and $(\{Y, C\}, t_4)$ in sequence; while P_3 is a 1-hop path consisting of a single edge $(\{X, C\}, t_5)$).

The charge to client and the reward to relays are determined by a *payment scheme* consisting of two algorithms, namely, a *payment set selection algorithm*, which decides which relays should be paid, and a *payment calculation algorithm*, which decides how much should be paid to each selected relay, and how much to charge the client.

As stated in Section III-A, MobiCent uses its message exchange protocol to constraint the strategy space of users, so that *edge insertion attack* and *edge hiding attack* are the two major forms of selfish actions that a node can take. We will illustrate how a selfish node gains from cheating under a natural payment scheme. The example is based on the contact graph in Figure 2. Without loss of generality, we assume the *earliest-path fixed-amount* payment scheme. Under the scheme, a client pays for each received data block a total amount of 3 cents, which is shared equally by all relays in the earliest delivery path. A helper participated if the payoff is more than 1 cent, thus the maximum path length is 3.

For illustration purpose, we redraw Figure 2 to highlight the edges that belong to different paths in Figure 3. Thus, some nodes (e.g., the client C), which are receivers in multiple edges, are plotted as multiple instances in the figure.

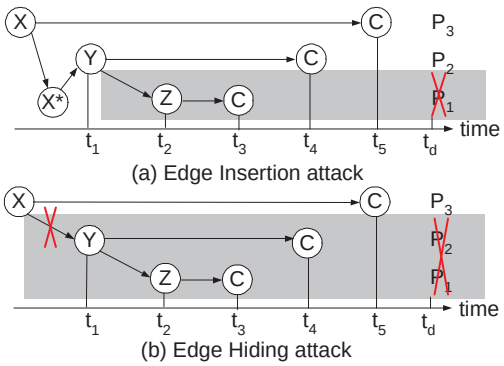


Fig. 3: Attacks

Figure 3 (a) shows an edge insertion attack. In the figure, when a selfish AP X gets the data, it estimates the delivery probability for all possible paths, denoted as $p(P_1)$, $p(P_2)$, and $p(P_3)$ respectively. Recall that the reward per node is $\frac{3}{n}$ cents where n is the hop count of the delivery path. Suppose $p(P_1) = 1$ and $p(P_2) = \frac{1}{2} + \epsilon (> 0)$. By creating a sybil of X^* and forging a phantom transfer from X to X^* before forwarding to Y, X can claim $\frac{2}{3}$ of the total payment if P_2 succeeds. However, due to this additional edge, Y will not be able to forward to Z, as the maximum hop 3 is reached already. Thus path P_1 is not revealed. By launching edge insertion attack, the expected payoff to X by forwarding via Y is $3 \times \frac{2}{3} \times p(P_2) = 1 + 2\epsilon$. In comparison, the payoff if X transfers honestly is only $3 \times \frac{1}{3} \times p(P_1) = 1$. As a result, the selfish behavior of node X increases its own payoff, but hurts the system performance by reducing the success delivery probability from 1 to as low as $\frac{1}{2} + \epsilon$ (if $p(P_3) = 0$), and the delay, if successfully delivered, is increased from t_3 to no less than t_4.

The client can also cheat by launching edge insertion attack. For example, when it meets X, it can pretend to be a relay instead, so that it can recover some of its payment as the sybil.

Figure 3 (b) shows an edge hiding attack. Depending on the estimated delivery probabilities, node X may decide not to forward the packet to other relays at all. Suppose $p(P_3) = \frac{2}{3} + \epsilon (> 0)$. In this case, in order to selfishly maximize its own reward, node X will not forward the data to Y, i.e., hiding the edge $(\{X, Y\}, t_1)$. Such an action has the same effect as dropping the packet. This holds regardless of the value of $p(P_1)$ and $p(P_2)$, and even when X is allowed to play edge insertion attack (as described above). The selfish behavior of node X hurts the system performance, by reducing the success delivery probability from up to 1, to as low as $\frac{2}{3} + \epsilon$, and increase the delay to t_5.

Given $G = (V, E)$, the two attacks can be formalized as:

Definition 4: **Edge insertion attack** of a node v is performed by creating a sybil v' such that G is modified to $G' = (V', E')$, where $V' = V \cup \{v'\}$, and $E' = E^{v \to (v,v')} \cup \{(v, v', t)\}$. $E^{v \to (v,v')}$ means for any edge e in $E(v)$, the vertex corresponding to node v can be set to either v or v'.

Definition 5: **Edge hiding attack** for a node v is performed by modifying G to $G' = (V, E - e)$, where $e \in E(v)$.

A cheater can launch one or both attacks multiple times. Now we can define the path revelation game formally.

Definition 6: A **path revelation game** is a distributed online game to reveal paths on a contact graph G.

- Each node (including both relay and client) is a player.
- As an edge e is formed, only its two vertex nodes together can reveal the existence of the edge. The possible strategies of a player are (1) acting honestly, (2) edge insertion attack, and (3) edge hiding attack.
- The payment scheme calculates payoff for each player based on the revealed contact graph.

The payment scheme determines the outcome of the game, and it should be designed to discover some desirable path from source(s) to destination (e.g., earliest revealed path or shortest revealed path). More specifically, we design payment schemes to meet the following goals:

1) *Incentive compatible*: Truthful participation is adopted by both relay and client, despite of their selfish nature.

2) *Efficient and frugal*: If there is at least one path revealed before a given deadline, the client should be able to recover the data with minimum payment. If a client is willing to pay more (but still bounded amount) to recover its data as soon as possible, the client should be able to recover its data upon revelation of the earliest path.

In the following, we first present the message exchange protocol to support MobiCent in Section IV. Following that, we analyze the payment algorithm required to combat edge insertion attack in Section V, then present the thwarting of edge hiding attack in Section VI.

IV. MobiCent Message Exchange Protocol

In *MobiCent*, we exploit the highly available low bit control channel to allow a Trusted Third Party (TTP) to mediate the file downloading/uploading process. We will explain the message flow using file downloading from Internet as an example. The case of a source node initiating a file transfer to a destination node is similar. Message exchanges occur in three stages: (1) data request; (2) data forwarding and (3) data recovery.

We assume that TTP's public and private keys are P_T and S_T respectively, while a participating node (helper or client) R's public and private keys are P_R and S_R respectively. In addition, R shares with TTP a symmetric key k_{TR}.

Each node only needs to know its own public and private keys, the shared secret key with TTP and TTP's public key. For the TTP, besides its own public and private keys, it has to know the shared secret keys and public keys of all the nodes. A new participating node has to inform TTP of its public key and choose the shared secret key with the TTP. Furthermore, TTP encrypts the pair $\{$*node id, node's public key*$\}$ with its private key and this signature is stored on the participating node.

A. Data Request

To initiate the downloading process, a client C first sends the file download request $r =< C, f, p(), t_0, t_d, \alpha >$ to TTP in a secure way. f is the file description including its name, size, and the approach to locate the file (e.g. URL address). $p()$ is the payment function, which will be discussed in detail in Section V. t_0 and t_d are the start time and deadline of the request respectively. α indicates the valid geographical area/region for the request to propagate.

After receiving and successfully decoding/verifying the request, TTP encrypts r with its private key and sends C the request signature $S_T(r)$.

Upon receiving TTP's approval, C can then forward $< r, S_T(r) >$ to all APs within the specified area α. C may need to contact a directory server to find out the list of APs in the area.

When an AP gets the request, it will first check the validity of the signature from TTP, as well as the file description, the time and area scope. It may also consider the amount of promised payment to decide whether to help or not. If the reward is sufficient, AP begins to prefetch the file block by block, with a predetermined block size. These blocks are then transmitted and replicated to helpers through the DTN.

B. Data Forwarding

Each node R maintains a list of blocks $\mathbb{L}_1(R)$ that it currently holds, and a list of blocks $\mathbb{L}_2(R)$ that it has requested but not received yet. When two nodes A and B are near each other, they can communicate directly via the high speed but short range radio. They will begin with an exchange of meta-data to reconcile their block list $\mathbb{L}_1(A)$, $\mathbb{L}_1(B)$, $\mathbb{L}_2(A)$ and $\mathbb{L}_2(B)$, and agree on the subset of blocks to be exchanged and the sequence to exchange blocks. The exact block subsets that are exchanged depend on the routing algorithm, for example, see [3] [4] [6].

For the i^{th} block of request r, the message being forwarded consists of three parts as shown in Figure 4 (a). The header H contains the basic information $< r, i, S_T(r) >$ which remains the same for all hops. The header is followed by the encrypted data and supplementary layers, which are being modified and appended to respectively at every hop.

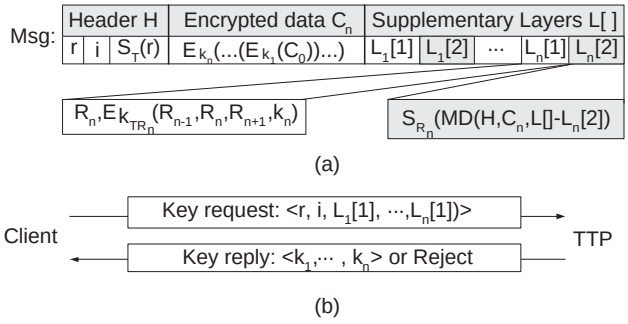


Fig. 4: Message format

Denote C_0 as the requested content in clear text, and C_n as the encrypted content forwarded by the n^{th} hop node ($n = 1, 2, ...$). Let the n^{th} hop relay be denoted by R_n.

Before forwarding a received block with data payload C_{n-1} to the next hop, the relay R_n generates a unique symmetric key k_n for the block, and substitutes the data payload with $C_n = E_{k_n}(C_{n-1})$. Note that k_n is only used to encrypt the current block and a new key is generated for each block encryption.

In addition, it appends a new supplementary layer with 2 components, $L_n[1]$ and $L_n[2]$. The first component $L_n[1]$ contains the current relay's ID R_n, and an encrypted field of four subfields, namely the previous relay's id R_{n-1}, the current relay's id R_n, the next relay's id R_{n+1} and the secret key k_n used for data block encryption. The shared secret key of TTP and R_n is used to encrypt this element. The array of $\{L_j[1]\}$ is the data that will

be forwarded to the TTP later by the mobile client to recover the data. For the source node, a randomly generated value is used for R_{n-1}.

The second component $L_n[2]$ consists of just one fields, a cryptographic hash (e.g. using MD5 or SHA-1) of the whole block minus the currently computed hash values, encrypted using the current relay's private key (S_{R_N}). This component is required for verification and auditing purpose and is only sent to the TTP when there is a dispute.

The next relay R_{n+1} first verifies the header H to make sure that request r is valid. Then, the relay R_{n+1} stores the data block and the identity R_n which is needed to generate the next supplementary layer if it forwards the message further. Before forwarding, it also needs to verifies $L_n[2]$ using R_n's public key. This key is verified using the TTP's signature for the pair $\{R_n, P_{R_n}\}$ obtained from R_n.

Note that a relay node does not need to contact TTP during the process. This has two benefits: (1) reduce the load of TTP; (2) enable a mobile node with only intermittent DTN link to become a relay.

In a forwarding process, for each block, a sender R_n needs to perform 2 symmetric key encryption (over the data payload and $L_n[1]$) and signs 1 fields ($L_n[2]$) using its own private key, and the receiver R_{n+1} needs to verify $L_n[2]$ using the sender's public key P_{R_n}. The receiver also need to verify the sender's public key (per neighbor overhead) and TTP's signature for the request (per request overhead).

C. Data Recovery

Without loss of generality, suppose the block is delivered from source R_1 to the client C via two store-and-forward hops $R_1 \rightarrow R_2$ with one time encryption key k_1, and $R_2 \rightarrow C$ with key k_2.

C sends to the TTP (in a secure way) the following key request $< r, i, L_1[1], L_2[1] >$ as shown in Figure 4 (b).

From this information, TTP is able to recover the required secret keys k_1 and k_2. TTP then sends $\{k_1, k_2\}$ to C.

With these keys, C is able to decrypt the data block using each key in the given list sequentially until all keys are used and the original text is recovered. At this point, we assume that C is able to validate clear text through checksum in the clear text or application level semantic. If data is valid, C sends confirmation to the TTP. Otherwise, C sends a dispute with the encrypted data it receives (C_n) and the list of elements in $\{L_n[2]\}$ to the TTP.

TTP settles the credit transfer off-line. Also, TTP may broadcast the ACK for block r, i in the area α after the request is completed.

If client does not submit any key request before the deadline, TTP will assume that the process fails. All pending data blocks in the network automatically time-out.

D. Protocol Properties

The message exchange protocol has the following properties. First, it prevents free-riding through the use of en-route onion encryption. There is no monetary barrier for a potential forwarder to participate. As the forwarder does not need to decrypt the data, it does not pay for the content.

Next, the protocol prevents a node from modifying an existing valid path segment since each relay encrypts the identities of the previous, current and next relays. Based on the information contained in the message, the protocol can deterministically detect nodes that modify the path.

Both TTP's communication and computation load are minimum. Relays do not need to contact TTP during forwarding, and payment settlement is performed offline. Finally, forwarding requires public-key cryptography which may be expensive. We discuss this issue further in Section VII-D.

V. Thwarting Edge Insertion Attack

Suppose relays on a delivery path are selected for payment, we consider the design of payment calculation algorithm to thwart edge insertion attack. We consider a general payment scheme S. Given a n hop path, we define the minimum payment to an individual relay in the path as $Reward_S^{min}(n)$, and define the charge to a client using a n hop path as $Charge_S(n)$.

***Lemma** 1:* To prevent a relay from gaining in edge insertion attack, $2 \times Reward_S^{min}(n+1) \leq Reward_S^{min}(n)$.

Proof: Consider the relay R earning the minimum payment in a n hop path, by inserting a sybil R', its reward is the sum of payments to two relays on a $n+1$ hop path, which is no less than $2 \times Reward_S^{min}(n+1)$. In order to prevent R from gaining by doing so, we must have $2 \times Reward_S^{min}(n+1) \leq Reward_S^{min}(n)$. ■

***Lemma** 2:* To prevent a client from gaining in edge insertion attack,

$Charge_S(n+1) \geq Charge_S(n) + Reward_S^{min}(n+1)$.

Proof: By appending a phantom edge on a n hop path, the client can gain reward as the sybil node. Since the new path contains $n+1$ hops, the reward to the appended sybil is no less than $Reward_S^{min}(n+1)$. In order to prevent client from gaining by doing so, $Reward_S^{min}(n+1) - Charge_S(n+1) \leq -Charge_S(n)$. ■

Note that, our formulation is general, as it does not exclude the use of other factors to determine payment. For example, we allow the rewards for different hops in a path to be different.

Lemma 1 states that the payment scheme should ensure that a relay's incremental gain by being paid as multiple sybils grows slower than the reduction of each individual's payment (due to the increase of path length). Similarly, Lemma 2 states that incremental increase of a client's payment for using a longer path is greater than the reward the client earns as the added sybil.

The two lemmas show that existing payment schemes, including the fixed-amount payment scheme we considered above, as well as the payment structure of [14] and [15] are not incentive compatible under edge insertion attack.

To simplify the presentation without loss of generality, we assume 1 cent is the minimum reward required to motivate a relay to participate in the forwarding process. Lemma 1 and Lemma 2 together lead to Theorem 1.

***Theorem** 1:* To enable incentive compatible forwarding while ensuring deficit-free for TTP[3], the payment charged to a client for using a n-hop path is at least $2^n - 1$.

Proof: As $Reward_S^{min}(n) \geq 1$, from Lemma 1, we have $Reward_S^{min}(i) \geq 2^{n-i}$ for $1 \leq i \leq n$. Using Lemma 2, we have: $Charge_S(n) \geq \sum_{i=1}^{n} Reward_S^{min}(i) \geq \sum_{i=1}^{n} 2^{n-i} = 2^n - 1$ ■

While the bound may seem large, we argue that it is feasible to be adopted in practice, because:

1) The client can specify the value of maximum hop N according to its requirement and utility function to bound the maximum possible payment.

[3]If the deficit-free property is not ensured, malicious node can make profit from phantom transactions.

2) While the cost of using a small N (3 to 5) is low, it is sufficient in most cases, as will be shown in Section VII.

As existing schemes do not satisfy the required property, we introduce a new incentive-compatible payment algorithm which minimizes the client's payment.

Multiplicative Decreasing Reward(MDR)

Given the maximum path length N and a small positive ϵ, if a n-hop ($1 \leq n \leq N$) path is selected, each relay on the path gets the same reward of:

$$Reward_{MDR}(n) = (2+\epsilon)^{N-n} cents \quad (1)$$

and the client is charged by

$$Charge_{MDR}(n) = (2+\epsilon)^{N} - (2+\epsilon)^{N-n} cents \quad (2)$$

***Theorem** 2:* Under MDR payment algorithm, both relays and client strictly have no incentive to launch edge insertion attack.

Proof: Under MDR payment algorithm, if a client on a n-hop path launches edge insertion attack, and inserts $k \geq 1$ extra edges, its net payoff is:

$$\begin{aligned} & k \times Reward_{MDR}(n+k) - Charge_{MDR}(n+k) \\ = \;& k \times (2+\epsilon)^{N-n-k} - ((2+\epsilon)^{N} - (2+\epsilon)^{N-n-k}) \\ = \;& \frac{k+1}{(2+\epsilon)^{k}}(2+\epsilon)^{N-n} - (2+\epsilon)^{N} \\ < \;& (2+\epsilon)^{N-n} - (2+\epsilon)^{N} \quad (since \quad \epsilon > 0, k \geq 1) \\ = \;& -Charge_{MDR}(n) \end{aligned} \quad (3)$$

Hence, a client does not gain by inserting edge. Now let us consider the last relay R_n on a n-hop path. Regardless of the behavior of previous relays (whether some of them are sybils or not), if R_n launches edge insertion attack and inserts k extra edges ($n < n+k \leq N$), its reward is:

$$\begin{aligned} & (k+1) \times Reward_{MDR}(n+k) \\ = \;& \frac{k+1}{(2+\epsilon)^{k}}(2+\epsilon)^{N-n} < Reward_{MDR}(n) \end{aligned} \quad (4)$$

Hence, the dominant strategy for R_n is to act truthfully. Similar argument can be applied iteratively to the other relays starting from the $n-1^{th}$ relay, assuming that later relays on the path are rational. Therefore, based on

iterative elimination of dominated strategy, all relays will eventually adopt truth-telling, which is the unique Nash equilibrium.

■

Note that truth-telling is not dominant strategy for relays except for the last relay since the strategy of a relay earlier in the path can be affected by an irrational relay later on the path. However, the game is dominance solvable and all relays adopt truth-telling in the unique Nash equilibrium.

Among payment schemes that satisfying the necessary conditions for incentive compatibility, Theorem 1 and Theorem 2 together imply:

Corollary 1: MDR payment algorithm is the most frugal incentive compatible mechanism robust under edge insertion attack.

Under the MDR payment algorithm, both relay's individual reward and the sum of all relays' reward decrease with the path length, while the client's payment increases with the path length. The maximum surplus or overpayment is reached when the longest path (N hops) is used, which is: $Charge_{MDR}(N) - N \times Reward_{MDR}(N) = (2+\epsilon)^N - (N+1)$.

This overpayment can be handled in the following ways. First, some of the overpayment can be considered as payment to the system provider. Second, the overpayment may be redistributed back to the mobile nodes if the redistribution is incentive compatible. An example of an incentive-compatible redistribution mechanism can be found in [28].

MDR alone is sufficient to handle edge insertion attacks given a selected set of relays. However, edge hiding attacks may affect the set being selected. Thus, MDR algorithm need to be used together with some payment set selection algorithm. In the following section, we present selection algorithms for two types of clients, namely:

- *Cost-sensitive client*: The client's goal is to minimize payment under a given deadline constraint.
- *Delay-sensitive client*: The client's goal is to minimize delay under a given payment constraint.

VI. Thwarting Edge Hiding Attack

The high-level idea to thwart edge hiding attack is to determine an incentive-compatible relay set by examining a sufficient subset of paths ever revealed before deadline.

A. Cost-sensitive Client

min-Cost Selection Algorithm Under this algorithm, the forwarding procedure is terminated only at deadline of the request, or upon revelation of a 1-hop path, whichever is earlier. Client reports to TTP the shortest path ever revealed when the terminating condition is met. Only relays on the reported path are paid. Payment by client and to relays are computed using the MDR algorithm.

***Theorem** 3:* Under min-Cost selection algorithm, both relay and client have no incentive to launch edge insertion attack or edge hiding attack.

Proof: We first consider the dominant strategy for the client. The client cannot arbitrarily fake a shortest path, as in that case it is not able to decode the correct data. Given that client pays the least with the shortest path it can reveal, it has no incentive to hide the shortest path it is able to get. Finally, Theorem 2 states that client has no incentive to append any sybil on the reported path.

For a given relay, we consider the two attacks sequentially:

1) Edge insertion attack: For a relay on the selected shortest path, Theorem 2 states that inserting edge on the selected path does not benefit the relay. Inserting edge on any non-selected path only increases its length, and does not make it the shortest path, thus, does not change the payment decision.

2) Edge hiding attack: for a relay on the selected shortest path P, hiding other paths do not have impact, and hiding the shortest path can result in two scenarios. First, another path that does not contain the relay is selected. Second, another path containing the relay but with length no shorter than P is selected. In both cases, the relay's payoff does not increase, hence there is no incentive. For a relay not on the shortest path, hiding any path that containing it does not affect the shortest path being selected, thus its payoff remains zero. ■

In Figure 5, all paths revealed to client are shown at their revelation time. The maximum path length $N = 3$. Note that client is not shown in the paths. Among all paths that are present, client only accepts P_1, P_3, and P_6, as each of them is the single shortest path at the moment they are revealed. Client reports the 1-hop path P_6 to TTP at t_6, as there is no path that shorter than it can be revealed. The client pays $Charge_{MDR}(1) = 2^3 - 2^{3-1} = 4$ cents, while relay U on the reported path is paid by $Reward_{MDR}(1) = 2^{3-1} = 4$ cents.

If the deadline t_d is between t_5 and t_6 instead, the client will report path P_3 at the new t_d. Relays Y and W on P_3 are paid, and each gets $Reward_{MDR}(2) = 2^{3-2} = 2$ cents, while client is charged by $Charge_{MDR}(2) = 2^3 - 2^{3-2} = 6$ cents. The surplus is $6 - 2 \times 2 = 2$ cents. Note that, there are multiple sources (node U and node Y) in this example.

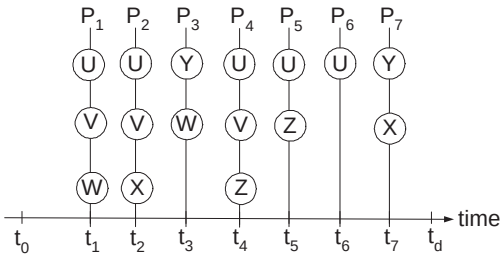


Fig. 5: Paths revealed over time axis

B. Delay-sensitive Client

In this case, the decryption keys for data are given to the client by TTP immediately when the earliest path is revealed. Designing incentive-compatible scheme for delay-sensitive client is more complicated than cost-sensitive client because payment to relays can only be finalized by examining the rest of the paths. Therefore, a mechanism

must be incorporated to motivate client to continue to reveal paths to TTP truthfully, even though it already has the decoded data.

Briefly, the **min-Delay Selection Algorithm** contains the following three steps:

1) Key revelation and initial payment by client: When the first path P_1 is revealed at t_1, the client immediately decrypts it through TTP, and is charged $n \times 2^{N-1} + (2^n - 2)$ cents, where N is the maximum path length, and n is P_1's hop count.

2) Reimbursement to client for reporting eligible paths: Clients continues to report eligible paths to TTP and client is reimbursed 1 cent for each *eligible path* it reports to TTP.

3) Payment set selection: Based on the eligible path sequence that the client reports, TTP decides the set of relays $\mathbb{R}$ to be paid. Once $\mathbb{R}$ is determined, MDR payment algorithm is applied over $\mathbb{R}$ to calculate the payment to relays.

We discuss the steps in more detail as follows:

Initial payment: In this step, the first portion of the payment $n \times 2^{N-1}$ prevents client from gaining by inserting a sybil in the earliest path and claiming back the maximum reward 2^{N-1} with the inserted sybil. The second portion of the payment $2^n - 2$ is the provident fund to pay the client for reporting eligible paths (maximum $2^n - 2$ paths with 1 cent each) in the next step.

For example, in Figure 5, the earliest path P_1 is used for decoding the message and calculation of client's initial payment. As $n = 3$, client pays $n \times 2^{N-1} + (2^n - 2) = 3 \times 2^{3-1} + (2^3 - 2) = 18$ cents.

Eligible path: Ideally, information about all paths can be collected. However, the number of paths can be unbounded. Furthermore, if there is no eligibility constraint on the path, client can fake any number of paths by appending its sybils on the earliest path or forging a path with only its sybils, to earn the reimbursement without receiving and reporting any real path. We define *eligible path* in the following way.

Definition 7: A path P is an *eligible path*, if and only if the intersection set of its relays and the relays on the earliest path P_1 is a *unique non-empty* subset of $Relay(P_1)$.

Uniqueness is defined in the following way. A path P is an eligible path if there is no other eligible path P' such that $Relay(P') \cap Relay(P_1) = Relay(P) \cap Relay(P_1)$.

The eligible path is defined to meet the following three conditions: (1) the size of the eligible path set must be bounded; (2) cheating from client cannot increase the eligible path set; and finally (3) TTP must be able to calculate an incentive compatible payment based on the eligible path set.

We illustrate the determination of eligible paths using Figure 5. Among all paths revealed after P_1, only path P_2, P_3, and P_5 are eligible. The total reimbursement to client for these three eligible paths is 3 cents. Paths P_4 and P_6 are not eligible paths due to the uniqueness constraint. Note that, client can hide P_2 to make P_4 an eligible path. However, doing this does not increase client's reimbursement. Finally, path P_7 is not an eligible path because its intersection set with P_1 is empty.

Payment set selection: Denote the initial payment set as $\mathbb{R}_1 = Relay(P_1)$. The payment set is updated every time an eligible path is revealed. The update rule is as follows. Suppose before an eligible path P is revealed, the

payment set is $\mathbb{R}_i$. If $\mathbb{R}_i \cap Relay(P) \neq \emptyset$, then the payment set is updated to $\mathbb{R}_{i+1} = \mathbb{R}_i \cap Relay(P)$. Relays in the final payment set $\mathbb{R}_k$ will be paid.

Let us look at the evolution of payment set in the example given by Figure 5. The eligible paths are $\{P_1, P_2, P_3, P_5\}$, and the initial payment set $\mathbb{R}_1 = \{U, V, W\}$. P_2 updates the payment set to $\mathbb{R}_2 = Relay(P_2) \cap \mathbb{R}_1 = \{U, V\}$. As P_3's intersection set with $\mathbb{R}_2$ is $\emptyset$, thus P_3 is not used. P_5 updates the payment set to $\mathbb{R}_3 = Relay(P_5) \cap \mathbb{R}_2 = \{U\}$, which is the final payment set. Thus, only relay U is paid, and the reward is $Reward_{MDR}(|\mathbb{R}_3|) = 2^{3-1} = 4$ cents.

Note that, the correct calculation of payment set using the above selection algorithm does not require the revelation of all eligible paths. However, reimbursing all eligible paths is important to prevent the client from manipulating the report. Otherwise, if TTP reimburses client only for eligible paths used in the computation, client may have the incentive to hide some eligible paths so as to increase the number of eligible paths needed. This will result in the incorrect (non incentive compatible) computation of the relay payment set.

We introduce a lemma before we present and prove the main theorem in this section.

***Lemma** 3:* Under the payment set selection algorithm specified above, suppose the payment set is $\mathbb{R}_i$ at time t, given a relay $R \in \mathbb{R}_i$, for every path $P \in \mathbb{P}^e(t)$, $R \in Relay(P)$ implies $\mathbb{R}_i - \{R\} \subset Relay(P)$.

Proof: We prove it by contradiction. Assume P^* is the earliest eligible path which is revealed before t and satisfies both $R \in Relay(P^*)$ and $\exists R' \neq R$ such that, $R' \in \mathbb{R}_i \quad \& \quad R' \notin Relay(P^*)$. Assume the payment set when P^* is revealed is $\mathbb{R}^*$. As P^* is revealed before $\mathbb{R}_i$. $\mathbb{R}_i \subseteq \mathbb{R}^*$, thus $R' \in \mathbb{R}^*$ as $R' \in \mathbb{R}_i$. We have $\emptyset \subset Relay(P^*) \cap \mathbb{R}_k$, as $R \in Relay(P^*) \cap \mathbb{R}_k$. We also have $Relay(P^*) \cap \mathbb{R}_k \subset \mathbb{R}_k$, as $R' \notin Relay(P^*) \cap \mathbb{R}_k$ but $R' \in \mathbb{R}_k$. P^* is the earliest path satisfying this condition, so it is an eligible path, and it is used to update the payment set to $Relay(P^*) \cap \mathbb{R}^*$, which results in the removal of R' from payment set, and causes contradiction. ■

***Theorem** 4:* Under min-Delay allocation algorithm, both client and relay have no incentive to launch edge insertion attack and edge hiding attack.

Proof: First, we show that client's dominant strategy is to act truthfully:

1) Edge insertion attack: By inserting a sybil in the earliest path (increasing its length from n to $n+1$), client need to pay an extra $[(n+1)2^{N-1} + (2^{n+1} - 2)] - [n2^{N-1} + (2^n - 2)] = 2^{N-1} + 2^n$ cents, while it can earn through the sybil by at most 2^{N-1} (if the sybil node is the single relay in the final payment set) plus 2^n cents (by reporting 2^n extra eligible paths). As the net payoff is non-positive, client has no incentive to insert sybil node in the earliest path. Inserting sybil node in latter paths does not change the eligible path set, thus does not benefit client either.

2) Edge hiding attack: Hiding the earliest path is against the client's goal which is to minimize the delay to recover data. Hiding latter eligible paths only reduce client's payoff. Thus, client has no incentive to hide path.

We now prove that relay's dominant strategy is to act truthfully too, by examining three types of relays in turn.

1) For a relay R in the final payment set $\mathbb{R}_k$: One one hand, creating sybil R' to launch edge insertion attack does not help, because: if R' is not in the final payment set, it does not earn R any extra reward. If R' is in the final payment set, the total amount earned by R and R' is $2 \times 2^{N-(|\mathbb{R}_k|+1)} = 2^{N-|\mathbb{R}_k|}$, which is equal to the reward

of having R alone. On the other hand, edge hiding attack does not benefit as well. If R is the only relay in the final payment set, it gets its optimal payment already. If $\mathbb{R}_k - \{R\}$ is not empty, using Lemma 3, all paths containing R also contains $\mathbb{R}_k - \{R\}$. Unless R eliminates itself from the final payment set, it can not exclude any node in $\mathbb{R}_k - \{R\}$ from final payment set either.

2) For a relay R not in the earliest path, inserting or hiding edge can not affect the revelation of the earliest path, thus does not bring it any reward.

3) Now let us consider a relay R in the earliest path, but is excluded from the final payment set. Without loss of generality, we assume R is eliminated from payment set $\mathbb{R}_{i-1}$ by a path P^*, i.e., $R \in \mathbb{R}_{i-1}$ but $R \notin \mathbb{R}_i$. Thus, P^* satisfies $\mathbb{R}_i \subset Relay(P^*)$ and $R \notin Relay(P^*)$. In addition, using Lemma 3, for every path P containing R that is revealed before P^*, $\mathbb{R}_i \subset Relay(P)$. Thus, to make itself appear in payment set before the revelation of P^i, R must make $\mathbb{R}_i$ appear in payment set also. In this case, P^* is always an eligible path to filter R out of the payment set. Even if R can hide all paths before P^i, P^i becomes the new earliest path, and it defines a new initial payment set which does not contain R at all. In this case, R still gets zero reward. Creating sybil does not prevent R (or any of its sybil) from being eliminated by path P^* either.

Thus, min-Delay algorithm is incentive compatible. ■

From Theorem 4, we directly have:

Corollary 2: min-Delay allocation algorithm reveals the earliest path, and client's payment is bounded by $O(N \times 2^N)$, where N is the longest forwarding path to be enabled.

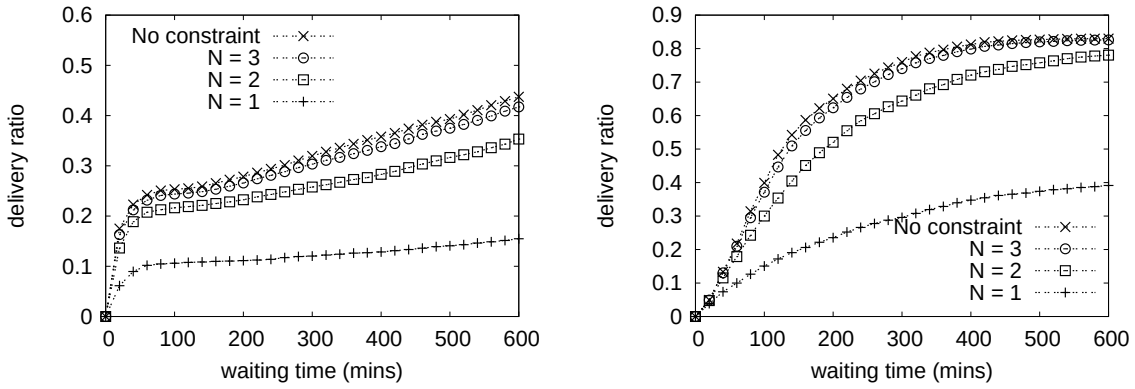
VII. Performance Evaluation

We evaluate *MobiCent* using the traces from the Haggle project [7] and DieselNet project [29], which represent human social networks and vehicular networks respectively. *MobiCent* treats the routing protocol as a black-box and is independent of the specific algorithm used. Our evaluation uses epidemic routing, and assumes each contact has sufficient capacity to exchange data. Performance under other routing protocols and constrained contact capacity show similar trends, and are not presented here to save space. Each experiment below is carried out 500 times with different seeds, and the average is presented.

We first evaluate the impact of hop count constraint on delivery performance. When all nodes are honest, we show that even if we set the maximum hop constraint N to a small value (3 to 5), the delivery performance already approximates the setting without constraint. Next, we evaluate the behavior of selfish nodes operating under the natural *earliest-path fixed-amount* payment scheme such that cheating may result in gains for some nodes. We show that cheating becomes the strategy of majority, and overall delivery performance degrades significantly. Payment schemes described in [14] and [15] have the same vulnerability, as none of them satisfy the properties we identified for incentive-compatible payment scheme in Section V. Lastly, we show the behavior of selfish nodes operating under *MobiCent*, and plot the resulted delivery performance as well as amount of payment by client.

A. Hop Count Limit

To evaluate the impact of hop count limit, we plot the delivery ratio over time where the maximum hop count is limited to 1 (direct delivery), 2 or 3, against the setting where there is no hop count constraint and all nodes are honest.



(a) Haggle trace (b) DieselNet trace

Fig. 6: Impact of hop count constraint

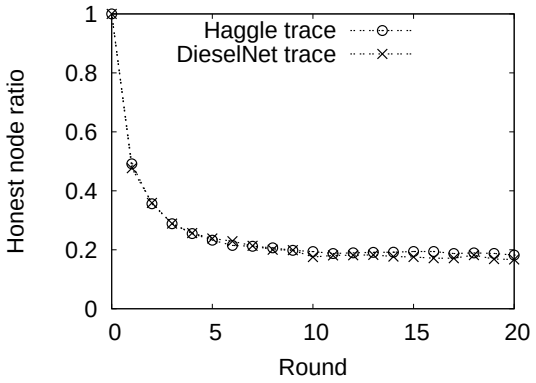
Figure 6 (a) plots the delivery ratio as a function of waiting time for Haggle trace under various maximum hop constraints of forwarding path. As shown in the figure, for any given deadline, the delivery ratio increases with the maximum hop count allowed. Allowing two-hop forwarding almost doubles the delivery performance of one-hop-only forwarding, while three-hop forwarding achieves more than 95% of the delivery ratio at any given deadline compared to the case without hop count constraint. Though not shown in the figure, five-hop forwarding achieves more than 99% of delivery performance. Similar result can be observed for DieselNet trace in Figure 6 (b). As a small N (≤ 5) suffices in most cases, the multiplicatively increasing payment of proposed schemes is practically affordable, as will be shown later.

B. Cheating under Earliest-path Fixed-amount Scheme

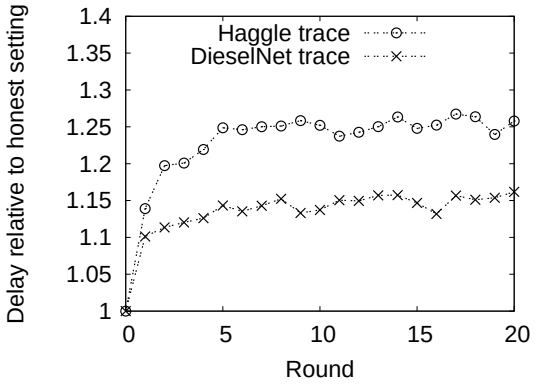
We study the user behavior under the earliest-path fixed-amount payment scheme, where a client pays a fixed amount (3 cents) to relays on earliest path for each block delivered. The amount is shared equally by all relays on the earliest forwarding path.

Figure 7 illustrates the system behavior when relays can cheat by hiding edges or creating sybils to increase their own payoff under both traces. In each round, one user generates two requests on average. In the first round, all nodes start truthfully. After each round, we assume each relay has access to the revealed contact graph and varies its strategy (acting truthfully or cheating) in the next round if it has a higher expected payoff with the new strategy based on its own past experience.

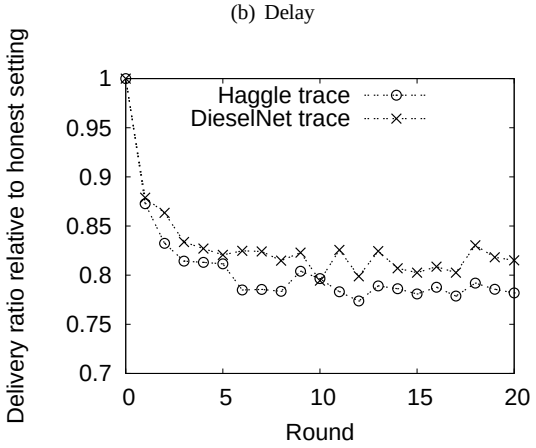
The nodes' behavior is shown in Figure 7 (a). Starting from a ratio of 100%, the ratio of honest users keeps decreasing and after 10 rounds, the system converges to a sub-optimal state. Note that, cooperation may still be



(a) Honest node ratio



(b) Delay



(c) Delivery ratio

Fig. 7: Evolution of user behavior under fixed-amount pricing scheme

preferred by some users (20%), as forwarding to other relay (honestly) increases the chance the node is in the selected path, which compensates the loss in having to share the reward with others.

Figure 7 (b) shows that the delivery delay increases under attack. The average delay is increased by 25% for Haggle trace, while it is increased by 15% for DieselNet trace. As shown in Figure 7 (c), delivery ratio decreases

by around 20% under attack for both traces.

Another way to measure the impact of dishonest nodes is to consider the relative gain of dishonest nodes vs. the honest nodes. When the ratio of dishonest nodes is fixed at 20%, simulation result shows that they collect more than 33% of the reward for both Haggle trace and DieselNet trace. The average reward of honest participants are reduced by around 20%, and is only around half the reward earned by cheating participants. When the ratio is increased to 50%, they collect 65% of the reward in Haggle trace and 75% of reward in DieselNet trace. In the latter trace, honest node's reward is reduced by 50%, and is only $1/3$ of the rewards of dishonest nodes. This indicates that a large portion of dishonest nodes can significantly decrease the reward for honest nodes. This has the effect of discouraging honest nodes from joining the system, further reducing the overall performance.

C. MobiCent Scheme Fosters Cooperation

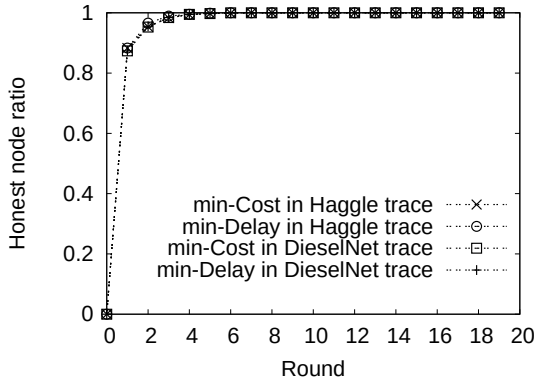
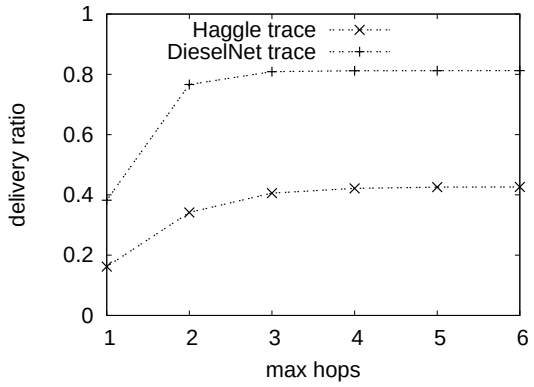


Fig. 8: Evolution of user behavior under MobiCent scheme

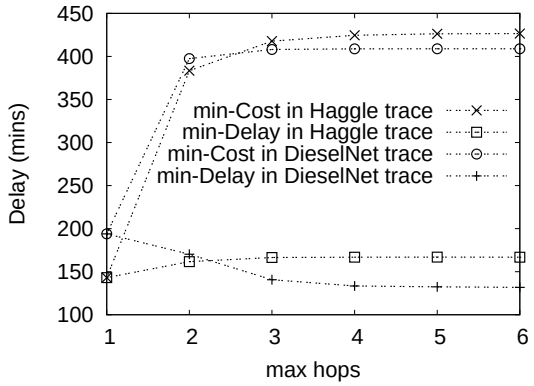
In order to evaluate how *MobiCent* fosters cooperation, we repeat the previous experiment but with all nodes initially cheating. As shown in Figure 8, from a state where all players cheat, and each player adapts its behavior based on its experience, all players converge to the truth-telling strategy very quickly, with 90% choosing to act truthfully after only 1 round. After 4 rounds, all nodes act truthfully and no node deviates from the truthful strategy any further. Such behavior applies to min-Cost and min-Delay schemes for both traces.

Figure 9 (a) shows the delivery ratios for the Haggle trace and DieselNet trace using the min-Delay and min-Cost algorithms. The delivery ratio is the same as the cases in which all nodes behavior honestly. This is expected since both of these algorithms ensure that there is no edge insertion and hiding attacks and should achieve the same behavior.

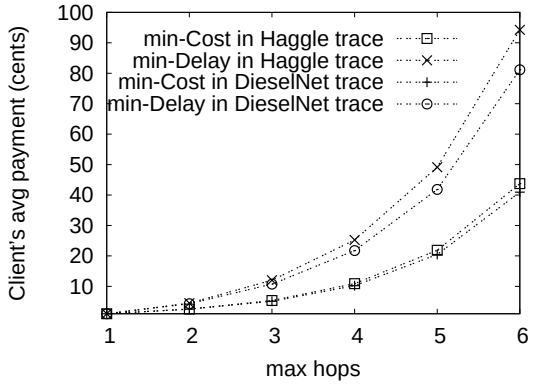
Figure 9 (b) plots the average delay for client to recover data under both schemes. The deadline is set to 600 minutes (10 hours). Since the first path received is reported in the min-Delay scheme, the delay is the same as the minimum achievable when all nodes are honest. When $N = 1$, the earliest path is also a single-hop path, thus the delay for both schemes are identical. When $N > 1$, min-Delay scheme still recovers data in the earliest path, while



(a) Delivery ratio



(b) Delivery delay



(c) Payment

Fig. 9: MobiCent performance under varying hop count constraint

min-Cost scheme needs to wait until the revelation of a single-hop path or the deadline, whichever is earlier. As shown in the figure, the delay for the min-Cost algorithm is more than double over the min-Delay algorithm. The client is compensated for this large increase in delay by having to pay less to the TTP.

Figure 9 (c) plots the average payment by client under both schemes. Recall that, as the maximum hop count N grows, the maximum payment grows at $O(2^N)$ and $O(N \times 2^N)$ respectively for min-Cost scheme and min-Delay scheme. The figure shows that the average payment grows in an exponential rate. However, as the average length of earliest path for both traces is around 2, the average payment by client under min-Delay algorithm is roughly two times of the average payment under min-Cost algorithm. Also recall that, when $N = 3$, the performance obtained is close to the case of no hop count constraint, in terms of both delivery ratio and delay. For $N = 3$, the average cost for min-Cost scheme is 5.36 cents under Haggle trace, and 5.10 cents under DieselNet trace, while the average cost for min-Delay scheme is 12.01 cents under Haggle trace, and 10.73 cents under DieselNet trace. Therefore, the payment is practically affordable based on the traces used, despite of the multiplicative growth.

D. Implementation Issues

We discuss two implementation issues, namely encryption key size and computation overhead.

There are two types of encryption keys. Public key encryption used is based on Elliptic Curve Cryptography (ECC) and 192-bit keys are used. The signature generated is 48 bytes. For symmetric key encryption, 128-bit AES algorithm is used. In order to reduce overhead, a 192-bit request identifier r_{id} can be selected and its signature computed by the TTP. These identifier and signature pairs can be used in the packet header instead of the original request string r. Assuming a 16KB data block and an average path hop count of 2, the average overhead imposed by the header and supplementary layer is about 250 bytes, less than 2% of the 16KB data block. Note that since the reward for breaking the *MobiCent*'s encryption is relatively small, the one time key size can be smaller in practice.

In order to evaluate the computation overhead, we measure the encryption and verification time of ECC on the target implementation platform, a Soekris Net5501 box. Using the OpenSSL library, measurements show that the average time for signing is 15ms and about 20ms for verification. The results show that these encryption schemes do not impose significant overhead. In fact, researchers have shown that it is viable to use public-key cryptography even on low power energy constraint platform using a 8-bit processor (Atml ATmegal128L), in particular, if ECC is used [30]. Finally, note that these encryption and verification tasks do not have to be performed in real-time and can be performed during the disconnected periods between contacts.

VIII. Conclusion

In this paper, we present *MobiCent*, a credit-based incentive system for DTN and prove that it is incentive compatible. *MobiCent* uses a Multiplicative Decreasing Reward (MDR) algorithm to calculate payment and supports two types of client, namely clients that want to minimize cost or minimize delay. Simulation results show that *MobiCent* can effectively foster cooperation among selfish nodes with bounded overhead.

References

[1] K. Fall, "A Delay-Tolerant Network architecture for challenged Internets," in *SIGCOMM*, Karlsruhe, Germany, August 2003.

[2] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks," CS-200006, Duke University, Tech. Rep., April 2000.

[3] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "MaxProp: Routing for vehicle-based disruption-tolerant networks," in *INFOCOM*, Barcelona, Spain, April 2006.

[4] A. Balasubramanian, B. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *SIGCOMM*, Kyoto, Japan, August 2007.

[5] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *WDTN*, Philadelphia, PA, USA, August 2005.

[6] B. Chen and M. Chan, "MobTorrent: A framework for mobile Internet access from vehicles," in *INFOCOM*, Rio de Janeiro, Brazil, April 2009.

[7] P. Hui, A. chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *WDTN*, Philadelphia, PA, USA, August 2005.

[8] N. Nisan, T. Roughgarden, E. Tardos, and V. V. . (editors)), *Algorithmic game theory*. Cambridge University Press, 2007.

[9] J. Douceur, "The sybil attack," in *IPTPS*, Cambridge, MA, USA, March 2002.

[10] E. Friedman and P. Resnick, "The social cost of cheap pseudonyms," *Journal of Economics and Management Strategy*, vol. 10, no. 2, pp. 173–199, 1998.

[11] U. Shevade, H. Song, L. Qiu, and Y. Zhang, "Incentive-aware routing in DTNs," in *ICNP*, Orlando, FL, USA, October 2008.

[12] FON, "wireless access provider," http://www.fon.com/en/info/makeMoney.

[13] L. Buttyan and J. Hubaux, "stimulating cooperation in self-organizing mobile ad hoc networks," *MONET*, vol. 8, no. 5, pp. 579–592, October 2002.

[14] M. Jakobsson, J. Hubaux, and L. Buttyan, "A micro-payment scheme encouraging collaboration in multi-hop cellular networks," in *Financial Crypto*, Gosier, Guadeloupe, January 2003.

[15] S. Zhong, Y. Yang, and J. Chen, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *INFOCOM*, San Francisco, CA, USA, March 2003.

[16] L. Anderegg and S. Eidenbenz, "Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents," in *MobiCom*, San Diego, CA, USA, September 2003.

[17] S. Zhong, L. Li, Y. Liu, and Y. Yang, "On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks — an integrated approach using game theoretical and cryptographic techniques," in *MobiCom*, Cologne, Germany, August 2005.

[18] H. Zhu, X. Lin, R. Lu, and X. Shen, "A secure incentive scheme for delay tolerant networks," in *Chinacom*, HangZhou, China, August 2008.

[19] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: Analysis & defenses." in *IPSN*, Berkeley, CA, USA, April 2004.

[20] C. Piro, C. Shields, and B. Levine, "Detecting the sybil attack in mobile ad hoc networks," in *SecureComm*, Baltimore, MD, USA, August 2006.

[21] Y. Du, R. Sami, and Y. Shi, "Path auction games when an agent can own multiple edges," in *NetEcon*, Ann Arbor, MI, USA, June 2006.

[22] Y. Sakurai, M. Yokoo, and K. Kamei, "An efficient approximate algorithm for winner determination in combinatorial auctions," in *EC*, 2000.

[23] A. Archer and E. Tardos, "Frugal path mechanisms," in *SODA*, San Francisco, CA, USA, January 2002.

[24] E. Elkind, A. Sahai, and K. Steiglitz, "Frugality in path auctions," in *SODA*, New Orleans, LA, USA, January 2004.

[25] A. Karlin, "Beyond VCG: Frugality of truthful mechanisms," in *FOCS*, Pittsburgh, PA, USA, October 2005.

[26] N. Mathewson, P. Syverson, and R. Dingledine, "Tor: the second-generation onion router," in *USENIX Security Symp.*, August 2004.

[27] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in *SAPIR Workshop*, Fortaleza, Brazil, August 2004.

[28] R. Cavallo, "Optimal decision-making with minimal waste strategyproof redistribution of VCG payments," in *AAMAS*, Hakodate, Hokkaido, Japan, March 2006.

[29] X. Zhang, J. Kurose, B. Levine, D. Towsley, and H. Zhang, "Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing," in *MobiCom*, Montreal, Qubec, Canada, September 2007.

[30] A. Wander, N. Gura, H. Eberle, V. Gupta, and S. Chang-Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," in *PERCOM*, Kauai Island, Hawaii, USA, March 2005.