Privacy-preserving Routing in Delay Tolerant Networks based on Bloom Filters

Evangelos Papapetrou*, Vasileios F. Bourgos*, and Artemios G. Voyiatzis†
* Department of Computer Science & Engineering, University of Ioannina
GR-45110, Ioannina, Greece
Email: epap@cs.uoi.gr
† SBA Research, Vienna, Austria
Email: avoyiatzis@sba-research.org

***Abstract*—Privacy preservation in opportunistic networks, such as disruption and delay tolerant networks, constitutes a very challenging area of research. The wireless channel is vulnerable to malicious nodes that can eavesdrop data exchanges. Moreover, all nodes in an opportunistic network can act as routers and thus, gain access to sensitive information while forwarding data. Node anonymity and data protection can be achieved using encryption. However, cryptography-based mechanisms are complex to handle and computationally expensive for the participating (mobile) nodes. We propose SimBet-BF, a privacy-preserving routing algorithm for opportunistic networks. The proposed algorithm builds atop the SimBet algorithm and uses Bloom filters so as to represent routing as well as other sensitive information included in data packets. SimBet-BF provides anonymous communication and avoids expensive cryptographic operations, while the functionality of the SimBet algorithm is not significantly affected. In fact, we show that the required security level can be achieved with a negligible routing performance trade-off.**

I. Introduction

Opportunistic networks, such as disruption and delay tolerant networks, can serve the needs for numerous application scenarios where continuous wireless connectivity among the network nodes cannot be guaranteed or is not possible at all [1]. In many cases, these networks are used to carry and transmit sensitive information from the field while the connectivity episodes are based on social interactions of the people carrying the nodes with them. Such an environment raises significant concerns regarding the security and the privacy these networks can offer and support. In this work, we focus on preserving privacy by *establishing node anonymity without affecting the routing operation*. That is, *two nodes can effectively use the routing mechanism to exchange data but no other node is able to reveal the identities of the communicating nodes*.

Earlier research efforts focus on providing anonymity in mobile ad hoc networks (MANETs) and peer-to-peer (P2P) networks [2],[3]. These approaches require the use of encryption (symmetric or public-key) and the definition of an end-to-end path between the sender and the receiver for a message delivery. Such knowledge of the global network topology is not available in opportunistic networks. The few published works focusing on opportunistic networks also require encryption and assume the on-demand access to a trusted key infrastructure.

In this work, *we opt for more efficient means of anonymity protection on opportunistic networks*. Our solution is based on algorithms that adhere to social network principles. This is a realistic assumption for mobile opportunistic networks connecting devices carried by humans. The human network of social contacts defines more or less the contact opportunities for their devices too. In the social-based routing algorithms, when two nodes get in reach of each other, they exchange information regarding their identity and their known contacts (neighbors). Based on this exchange, they decide which packets to pass to the other end for further forwarding. We aim to protect this information so as not to reveal the identities of the participating nodes and their neighbors and thus, enhance the anonymity of the system.

Our proposal is the SimBet-BF algorithm, an improved version of the SimBet algorithm that utilizes Bloom filters for achieving node anonymity. The algorithm does not use cryptography but only low cost operations on Bloom filters. This is a significant advantage for mobile devices with resource constraints (e.g., processor and battery). We experimentally prove that the packet delivery rate and the delivery delay is almost equal to that of the original SimBet algorithm.

The rest of the paper is organized as follows. Section II reviews the DTN and opportunistic routing algorithms with emphasis on social-based ones, analyzes the notion of network privacy, the challenges faced in networks consisting of mobile nodes, and the use of Bloom filters in networking. In Section III-B, we formulate the problem of anonymity in the context of opportunistic routing. Section IV describes the proposed solution, SimBet-BF, while Section V analyzes the achieved privacy. Then, in Section VI, we experimentally evaluate the performance of SimBet-BF. Finally, Section VII concludes our paper and presents future directions of research.

II. Related Work

A. DTN and opportunistic routing

The DTN routing can be classified in three main categories: flooding-based, history-based, and social-based. The flooding-based algorithms are among the most simplistic. Epidemic Forwarding is among the most well-known examples [4]. There, each node produces one copy of each message each time it meets another node and forwards the message to the node. If a node already holds a copy of the message, the forwarding does not occur. This approach drains the resources of the network, including node buffers, memory, and available

bandwidth, since multiple copies of the same message flood the network and network-wide cleanup activities must take place once the message is delivered to its final destination. In order to reduce the network load, some variations of the Epidemic Forwarding were proposed, such as PREP [5] and Spray-and-Wait [6]. These algorithms try to limit the number of copies in the network by introducing constraints on the number of hops or the lifetime of a message copy.

The history-based algorithms aim at improving the resource usage by exploring the contact history of each node. In this setting, if a node has met a lot of times another node in the past, it is expected that they will meet again in the future. The PRoPHET algorithm uses this information to compute a delivery probability for each message upon each contact and decide which node is the more probable to deliver each message at its destination [7].

The social-based algorithms make the observation that the human relationships are reflected in the contact opportunities of the devices they carry with them. In this case, the routing decisions are based on metrics coming from Social Analysis and characterize the importance of each node on the network. Some well-known algorithms of this category are the Bubble Rap [8] and the SimBet [9].

In order to improve delivery efficiency, multicopy versions of both history-based and social-based algorithms have been proposed along with methods for minimizing the number of replicas [10].

B. Network privacy

There exist proposals in the literature for supporting network privacy in the case of fixed infrastructure and end-to-end communication. However, in the case of mobile ad hoc networks, not only access points but rather each and every node of the network is a potential router for the messages and thus, it can break the node anonymity.

The most common type of attacks on network privacy is the traffic analysis. The simplest traffic analysis attack is packet sniffing from the shared medium, revealing the source and the destination node of each message. Packet tracing follows the packet flow within the network and reveals communication patterns among the nodes. A TTL-based attack correlates the temporal locality (as measured by the time-to-live field in the packets) with the proximity of a node to the message sender and destination.

The defense mechanisms are based on the use of cryptography. A well-known example is the layered message encryption in the Onion Routing [11]. This technique encrypts the message with the cryptographic key of each intermediate node it should pass. It requires the a priori definition of the path the message must follow in order to reach its final destination. The use of cryptography in multiple layers ensures that no intermediate node can read the message contents, including its source and its destination. The Onion Routing cannot be applied practically in opportunistic and mobile ad hoc networks, since the message path cannot be defined a priori.

Choi et al. [12] proposed the use of pseudonyms based on secret keys provided by a trusted authority. The pseudonym of each node must be changed frequently and the communication with the trusted authority should always be possible.

A third approach is to use multiple paths for each destination so as to diffuse the message segments through different paths and harden traffic analysis [13]. This approach is not suitable for opportunistic networks since defining and using one or more end-to-end paths is not possible.

Earlier attempts on DTNs and opportunistic networks include the work of Kate et al. [14]. This assumes the existence of a trusted third party acting as a public key generator for identity-based cryptography (IBE) [15]. There, the nodes can produce pseudonyms by their own. The method ensures anonymity only at the first and last hop of the communication. Also, access to the trusted third party must be assumed for distributing new keys and invalidating old ones as nodes enter and leave the network.

The SPRING protocol can be used in vehicular DTN [16]. SPRING assumes vehicular nodes (e.g., cars) and trusted roadside units (RSUs), which assist the routing operations. Geographical coordinates are used as pseudonyms instead of node identities and the vehicles encrypt their messages using public-key cryptography. The use of geographical coordinates ensures only a weak form of anonymity, since the presence of one or a few vehicles in an area can be easily correlated with senders and receivers at that area. Also, it is desirable to avoid, if possible, the use of heavyweight cryptography. Finally, the existence of trusted nodes (RSUs) that will not attempt to reveal node identity can work in a vehicular DTN but cannot be generalized in the case of opportunistic networks.

C. Bloom filters for routing and anonymity

A Bloom filter is a space-efficient probabilistic data structure that is used to represent sets [17]. The properties of Bloom filters are useful for network communication. As a result, Bloom filters have been used in various network applications [18],[19],[4]. An example of using Bloom filters in opportunistic networks is the case of Epidemic Forwarding. When two nodes encounter each other, they exchange the information stored in their buffers. In order to avoid useless transmissions and control the redudancy, nodes can exchange a summary vector indicating the packets already received [4]. A Bloom filter can substantially reduce the space overhead for representing a summary vector since it suffices for a node to know whether a message has been received before rather than its exact identity (i.e., it is sufficient to perform a membership check on the Bloom filter).

A novel solution for network privacy that avoids the use of heavyweight cryptography is proposed in [20] and then improved in [21]. The authors present a threat model for social-network routing and three versions of a privacy-enhanced social network routing: Statistically-manipulated (SSNR), Obfuscated (OSNR), and a combined SSNR-OSNR scheme. Each node maintains a list of its "friends" and selects a priori a set of nodes from this list. It uses only nodes from this set so as

to forward the pending messages. The proposed schemes aim to protect the friend list of each node.

In the case of SSNR, the friend list is manipulated for each message so as to exclude some friends and to incorporate some fake ones. In the case of OSNR, the friends list is embedded in a Bloom filter and each node can only perform a membership check on the received list. OSNR does not provide perfect security. The Bloom filter is constructed using the real addresses of the network nodes. These addresses are public information and known by all network nodes. A malicious node can launch a brute force attack and check if each and every node of the network belongs to the forwarders list of a node. Finally, SSNR-OSNR combines the two schemes into a new one: first the SSNR is applied for deriving the friends list and then the OSNR embeds the information in a Bloom filter.

III. Problem Formulation

In this work, we are concerned with preserving privacy by providing two nodes with the ability to communicate anonymously. That is, *two nodes can communicate while no other node is able to deduce their identities*. As mentioned earlier, the routing protocols in opportunistic networks follow the store-carry-and-forward approach. Packets are exchanged on an "encounter" (or "contact") basis by utilizing the so called *encounter protocol*. Therefore, *our task for achieving anonymous communication is to anonymize the encounter protocol*.

An encounter protocol ensures that the nodes in contact: a) agree on the list of packets that are candidates for forwarding, b) exchange the required "routing information", and c) exchange data packets according to the routing information. The implementation details of the encounter protocol depend on the actual routing protocol. Furthermore, each protocol defines "routing information" in a different way. In most protocols, routing information consists of one or more utility metrics that are defined by the algorithm. However, in many algorithms [9],[22], including most social-based ones, instead of exchanging utility metrics, the nodes exchange their lists of past encounters and use this information to calculate some kind of a utility metric. We examine anonymity in the context of this category of algorithms. This is a more challenging task because anonymizing the encounter protocol also involves the anonymous representation of the node's past encounters.

In the following, we will focus on the SimBet algorithm [9]. SimBet is one of the most representative algorithms of this category and probably the most demanding in terms of the operations needed to extract the utility from the lists of encounters. Nevertheless, *we feel that the proposed method could be easily implemented to any algorithm that uses information about encounters*. We further discuss this issue in Section IV-G. Before delineating the challenges for achieving anonymity, we briefly describe the SimBet algorithm.

A. Fundamentals of the SimBet Algorithm

The SimBet algorithm combines two utility metrics, the *ego betweenness centrality* and the *similarity*. Each node u maintains information for up to two-hop contacts: C_u is the list of node's encounters and C_u^2 is the list of encounters of the node's encounters (two-hop contacts) i.e., $C_u^2 = \{C_v : v \in C_u\}$. The node uses these information so as to calculate the two metrics.

The ego betweenness centrality of u is derived from its *betweennes centrality*. In fact, there exist multiple variations of centrality. In general, a centrality metric measures the importance of the node for the network: a node with a high centrality contributes more to the coherence of the network and thus, it is considered more probable to deliver a message to its destination. The *betweennes centrality* (or betweenness) of u depends on the total number of the shortest paths between network nodes that u is part of. Let $g_{j,k}$ denote the number of shortest paths among any node pair j, k and $g_{i,k}(u)$ the number of those paths that also include u. Then, the betweenness centrality of u is defined as:

$$BC(u) = \sum_{j=1}^{N} \sum_{k=1}^{N} \frac{g_{j,k}(u)}{g_{j,k}} \tag{1}$$

The ego betweenness centrality of u is a local scope version of its betweenness centrality [23] i.e., it is calculated in its *ego network*. The latter is a network where the set of vertices includes u and its contacts C_u. An edge between two vertices indicates that the corresponding nodes have encountered each other. Note that the ego network can be constructed using C_u and C_u^2.

The similarity of a node u with a node v is the number of their common contacts i.e., $|C_u \cap C_v|$. The idea is that if node u shares a lot of contacts with node v, then it is very probable that they will meet each other in the future, even if they have not done so already. Also, if a node v shares a lot of neighbors with the destination node of the message, the same line of thinking suggests that it is more probable that node v will manage to deliver a pending message to its final destination.

SimBet uses a typical example of an encounter protocol when two nodes e.g., A and B, meet each other. Let Bet_A and Bet_B denote the corresponding ego betweenness values while $\text{Sim}_A(i)$ and $\text{Sim}_B(i)$ denote the similarity of A and B with node i, respectively. The utility metric used for exchanging messages during the encounter protocol (SimBetUtil) is a linear combination of ego betweenness and similarity. More specifically, the $\text{SimBetUtil}_A(i)$ of A for node i is computed using the following expressions:

$$\text{SimUtil}_A(i) = \frac{\text{Sim}_A(i)}{\text{Sim}_A(i) + \text{Sim}_B(i)}$$

$$\text{BetUtil}_A = \frac{\text{Bet}_A}{\text{Bet}_A + \text{Bet}_B}$$

$$\text{SimBetUtil}_A(i) = \alpha \cdot \text{SimUtil}_A(i) + \beta \cdot \text{BetUtil}_A$$

where $\alpha = \beta = 0.5$. The encounter protocol involves the following steps:

(1) As the nodes roam, they broadcast `hello` messages with their identity, as to announce their presence.

(2) When A detects node B, the former delivers any pending messages that have B as their destination.
(3) A requests the list of encounters of node B.
(4) B replies with its list of encounters C_B (*encounter vector*).
(5) A, after updating its ego network, computes Bet_A and the similarity for the destination of each message it holds.
(6) A sends a *summary vector* to B. This contains Bet_A and a tuple $<i, \text{Sim}_A(i)>$ for each distinct node i for which A has messages to deliver.
(7) B computes Bet_B and the similarity for the destinations included in the summary vector. Then, it combines the two metrics so as to derive the SimBetUtil for both A and B.
(8) B sends a message *request vector* to node A. The vector contains all the destination identities i for which $\text{SimBetUtil}_B(i) > \text{SimBetUtil}_A(i)$.
(9) Finally, node A sends to node B all the requested messages, i.e., the messages that their destination is included in the request vector.

B. Challenges for an Anonymous Encounter Protocol

The description of the SimBet encounter protocol reveals that the identity of a node is included in several types of messages: `hello`, data, summary vector, request vector, and in C_A and C_B. Consequently, anonymizing the encounter protocol requires the:

- *anonymity of the sender and the receiver of a data message*: nodes A and B should not be able to deduce the identity of the sender and the receiver of any packet that they carry.
- *anonymity of the identities of the nodes in contact*: node A should not be able to infer node's B identity (and vise-versa) so as to eliminate the possibility of compromising privacy by intercepting packets destined to B in the future.
- *anonymous representation of contact information, i.e., $\mathcal{C}_A$ and $\mathcal{C}_B$*: this is for eliminating the possibility that a node can combine encounter lists from different nodes in order to reveal the identity of another node.

Note that providing anonymity should not hamper the efficiency of the routing mechanism. In other words, each packet should be correctly delivered to its destination using the same path that would be used by the non-anonymous routing mechanism. This implies that the best forwarder is selected in each hop. In the case of the SimBet, this is equivalent to correctly calculating both the ego betweenness and the similarity, even if an anonymous representation of contact information is used.

In the next section, we propose and describe SimBet with Bloom filters (SimBet-BF) that aims to provide complete network privacy.

TABLE I
SIMBET-BF NOTATION AND DEFINITIONS

$\mathcal{C}_i$	The list of node i's encounters
$\mathcal{C}_i^2$	The set of node i's two-hop encounters
$K_{i,j}$	The key used in the one-way communication where the sender is i and the receiver is j
K_i	A special unique key for node i
$\text{BF}(k)$	A Bloom filter containing element k (or every element in k if k is a set of elements)
dst_i^j	A Bloom filter used by i as the destination address when sending messages to j
src_i^j	A Bloom filter used by i to decide whether the sender of a message is j
id_j^i	A Bloom filter used by i for representing j as its encounter
h_i	A Bloom filter used as the identity of i in its `hello` messages

IV. THE ANONYMOUS SIMBET ALGORITHM

SimBet-BF is a social-based routing algorithm for opportunistic networks that ensures network privacy through the use of Bloom filters. It avoids the use of heavyweight cryptography so as to fit the operating environment of infrastructureless, opportunistic networks. In the following, we briefly describe Bloom filters and then proceed with the operations of the SimBet-BF algorithm. Table I summarizes the notation and definitions that will be used hereafter.

A. Preliminaries

A Bloom filter is a space-efficient probabilistic data structure that is used to represent sets and provides fast addition and membership test operations [17]. An empty Bloom filter is a bit array of m bits, all set to zero. To construct the filter, k hash functions are used. Each of them maps an input element to one of the m array positions. To *add* an element to the array, one feeds it to the k hash functions to get k array positions and sets the bit positions to one. To *check* membership, one follows the same procedure and compares the result with the bit array. If at least one of the positions that result from the k hash functions is set to zero in the filter, then the element is *definitely* not in the set. If all positions are set to one, then the element is *probably* a member of the set. This means that there is a *false positive* probability, i.e., a probability that the element is not in the set despite the result of the membership test. The false positive probability is given by:

$$P_{fp} = (1 - (1 - \frac{1}{m})^{k \cdot n})^k \quad (2)$$

where n is the number of elements in the set. Note that the union and intersection operations on two or more Bloom filters can be easily computed as the bitwise-OR and bitwise-AND of the input Bloom filters.

B. Bootstrap process

We assume the existence of a trusted authority (TA) that can produce, in an *off-line* mode, keys for the nodes of the network. This bootstrap assumption is common in the literature. Let $\mathcal{N}_r$ be the set of the real addresses of the nodes of the network and $\mathcal{N}_f$ be a set of unused addresses of fake (non-existent) nodes of the network. Let $\mathcal{N} = \mathcal{N}_r \cup \mathcal{N}_f$ be the set of all addresses handled by the TA, while $\mathcal{N}_r \cap \mathcal{N}_f = \emptyset$. Denote

TABLE II
KEYS CREATED BY THE TA IN AN EXAMPLE NETWORK ($|\mathcal{N}| = 7$, $\mathcal{N}_r = 5, \mathcal{N}_f = 2$)

Nodes	Real ($\mathcal{N}_r$)					Fake ($\mathcal{N}_f$)	
↱	A	B	C	D	E	F	G
A	K_{A}	K_{AB}	K_{AC}	K_{AD}	K_{AE}	K_{AF}	K_{AG}
B	K_{BA}	K_{B}	K_{BC}	K_{BD}	K_{BE}	K_{BF}	K_{BG}
C	K_{CA}	K_{CB}	K_{C}	K_{CD}	K_{CE}	K_{CF}	K_{CG}
D	K_{DA}	K_{DB}	K_{DC}	K_{D}	K_{DE}	K_{DF}	K_{DG}
E	K_{EA}	K_{EB}	K_{EC}	K_{ED}	K_{E}	K_{EF}	K_{EG}
F	K_{FA}	K_{FB}	K_{FC}	K_{FD}	K_{FE}	K_{F}	K_{FG}
G	K_{GA}	K_{GB}	K_{GC}	K_{GD}	K_{GE}	K_{GF}	K_{G}

TABLE III
BLOOM FILTERS PROVIDED BY THE TA TO NODE A

id_A^B=BF(K_{CA},K_{DA},K_{EA},K_{FA},~~K_{GA}~~,K_{B})↔id_B^A=BF(K_{CB},K_{DB},K_{EB},K_{FB},~~K_{GB}~~,K_{A})			
id_A^C=BF(K_{BA},K_{DA},K_{EA},~~K_{FA}~~,K_{GA},K_{C})↔id_C^A=BF(K_{BC},K_{DC},K_{EC},~~K_{FC}~~,K_{GC},K_{A})			
id_A^D=BF(K_{BA},K_{CA},K_{EA},K_{FA},~~K_{GA}~~,K_{D})↔id_D^A=BF(K_{BD},K_{CD},K_{ED},~~K_{FD}~~,K_{GD},K_{A})			
id_A^E=BF(K_{BA},K_{CA},K_{DA},~~K_{FA}~~,K_{GA},K_{E})↔id_E^A=BF(K_{BE},K_{CE},K_{DE},K_{FE},~~K_{GE}~~,K_{A})			
id_A^F=BF(~~K_{BA}~~,K_{CA},K_{DA},K_{EA},K_{GA},K_{F})↔id_F^A=BF(K_{BF},K_{CF},~~K_{DF}~~,K_{EF},K_{GF},K_{A})			
id_A^G=BF(K_{BA},~~K_{CA}~~,K_{DA},K_{EA},K_{FA},K_{G})↔id_G^A=BF(~~K_{BG}~~,K_{CG},K_{DG},K_{EG},K_{FG},K_{A})			
B→	dst_A^B=BF(K_{AB}),src_A^B=BF(K_{BA})	E→	dst_A^E=BF(K_{AE}),src_A^E=BF(K_{EA})
C→	dst_A^C=BF(K_{AC}),src_A^C=BF(K_{CA})	F→	dst_A^F=BF(K_{AF}),src_A^F=BF(K_{FA})
D→	dst_A^D=BF(K_{AD}),src_A^D=BF(K_{DA})	G→	dst_A^G=BF(K_{AG}),src_A^G=BF(K_{GA})
h_A=BF(K_{B},K_{C},K_{D},K_{E},K_{F},K_{G})			

with $K_{ij} : i, j \in \mathcal{N}, i \neq j$ the key of the communication with sender i and destination j and with K_{ji} the key for the reverse direction. In general, $K_{ij} \neq K_{ji}$ so as to distinguish the two directions of communication. Also, denote with $K_i : i \in \mathcal{N}$ a special, unique key for each node. Finally, let $\mathcal{N}_i = \mathcal{N} - \{i\}, \forall i \in \mathcal{N}$ be the set of all addresses handled by the TA excluding node i. The TA produces $|\mathcal{N}|^2$ keys in total. Table II presents the keys produced by the TA in an example network with $|\mathcal{N}| = 7$ ($\mathcal{N}_r = 5$ and $\mathcal{N}_f = 2$).

The TA never distributes the keys to nodes. Rather, it constructs and distributes Bloom filters (BF) for each node i as follows:

- *Destination identities*: A Bloom filter $dst_i^j = \mathrm{BF}(K_{ij}), \forall j \in \mathcal{N}_i$. This filter is used as the destination address in a data message sent from node i to node j.
- *Source identities*: A Bloom filter $src_i^j = \mathrm{BF}(K_{ji}), \forall j \in \mathcal{N}_i$. When node i receives a message, it uses src_i^j to decide whether j is the sender of the message. Observe that $dst_i^j = src_j^i$.
- *Node identities*: A Bloom filter $id_j^i = \mathrm{BF}(\{K_i\} \cup \{K_{vj} : v \neq i, j\}), \forall j \in \mathcal{N}_i$. The set corresponds to the j-th column of Table II excluding the special key K_j and replacing K_{ij} with the special key K_i. The filter id_j^i is the identity of j as perceived by node i. As will be discussed in Section IV-C, the filters $\{id_j^i : j \in N_i\}$ are used in the construction of the list of encounters ($\mathcal{C}_i$) and the set of two-hop encounters ($\mathcal{C}_i^2$). The TA also gives to node i the filters $\{id_i^j : j \in N_i\}$.
- *Hello identity*: The TA gives to node i the Bloom filter $h_i = \mathrm{BF}(\{K_j : j \neq i\})$. This corresponds to the main diagonal of Table II, excluding the key K_i. This filter serves as the identity that should be included in the `hello` messages sent by node i.

Once a node receives the appropriate identities, it can enter the network.

For the sake of security, the TA makes a slight modification of the aforementioned procedure. When creating a node identity for a real node, a fixed number, $x : 0 < x < |\mathcal{N}_f|$, of keys that are related to fake nodes are not inserted in the Bloom filter. The choice of which keys to exclude is random. For example, in the case of $x = 1$ and based on the example of Table II, the key K_{GA} may be omitted when constructing id_A^B and the key K_{FA} may be omitted when constructing id_A^C. When creating a node identity for a fake node, real keys can be also omitted. This is not an issue, since fake nodes should never exist in the network. The security enhancement of this procedure will be discussed in Section V. Table III illustrates an example of the Bloom filters provided to node A by the TA ($x = 1$).

The described bootstrap process ensures that:

- No actual address (e.g., A) or key information (e.g., K_{A}) is ever delivered to any node of the network.
- No two nodes get the same source and destination identities for a (real) node j i.e., $dst_i^j \neq dst_k^j \forall i, k \in N$ and $src_i^j \neq src_k^j \forall i, k \in N$.
- No node knows whether a destination address refers to a real (existing) node or a fake (non-existing) node.

C. Encounter protocol & structures

After the bootstrap process, the nodes can start exchanging data messages. In case a node i wishes to send a message to j then it should place dst_i^j as the destination address. Actually, this filter serves also as the source address. This is because node j can use its pool of source identities ($dst_j^k, \forall k \in \mathcal{N}_j$) to determine that the sender of the message is node i (recall that $src_j^i = dst_i^j$).

Data messages are forwarded on a encounter basis. To enable the detection of encounters, a node i periodically broadcasts `hello` messages that contain the filter h_i. A receiving node j can detect the encounter with node i by checking the received filter h_i against its pool of node identities (i.e., id_j^i) received from the TA during the bootstrap process. Using the example of Table II and assuming $i = B$ and $j = A$, then $h_B = \mathrm{BF}(K_{\mathrm{A}}, K_{\mathrm{C}}, K_{\mathrm{D}}, K_{\mathrm{E}}, K_{\mathrm{G}})$. It suffices for node A to perform a bitwise-AND between h_B and $id_A^x, \forall x \in \mathcal{N}_A$ so as to decide that the encountered node is related to $id_A^B = \mathrm{BF}(K_{\mathrm{CA}}, K_{\mathrm{DA}}, K_{\mathrm{EA}}, K_{\mathrm{FA}}, K_{\mathrm{B}})$. This is because h_B and id_A^B share no common key while each id_A^x shares with h_B the key K_x. Note that this operation does not reveal the identity of B to A (see a detailed discussion in Section V) but allows A to build an anonymized encounter list by adding $id_B^A = \mathrm{BF}(K_{\mathrm{CB}}, K_{\mathrm{DB}}, K_{\mathrm{EB}}, K_{\mathrm{FB}}, K_{\mathrm{A}})$ (but not id_A^B) to its encounter list $\mathcal{C}_A$. In general, a node i builds its encounter list using the node identities of the encountered nodes, i.e., $\mathcal{C}_i = \{\{id_x^i\}, x \in N_i\}$. Note that $\mathcal{C}_i$ is also a Bloom filter and adding a node identity id_x^i requires a bitwise-AND operation between $\mathcal{C}_i$ and id_x^i.

Continuing with the example of an encounter between A and B, the encounter protocol of SimBet-BF is as follows:

(1) As the nodes roam, they broadcast `hello` messages with their hello identity, h, so as to announce their presence.
(2) When A receives a `hello` message from B, it requests the encounter list of node B.
(3) B sends its list of encounters encoded as a Bloom filter that contains the identity id_x^B for each encounter x.
(4) A updates its encounter list and its two-hop encounters and computes its betweenness (Section IV-E) and its similarity with the destination of each message it holds. This is done using the destination identity that is recorded in the message (Section IV-D).
(5) A sends a summary vector to node B. The vector contains Bet_A and a tuple $<\text{dst}_i^j, \text{Sim}_A(\text{dst}_i^j)>$ for each distinct destination dst_i^j.
(6) B computes its betweenness and its similarity with the destinations included in the summary vector. It combines them so as to derive the SimBetUtil for both A and B.
(7) B sends a message request vector to node A containing all the destination identities dst_i^j for which node B has a greater SimBetUtil.
(8) Finally, A sends to node B all those data messages that have a destination included in the request vector.

Note that, in Step 4 of the encounter protocol, node A updates its encounter list as well as its two-hop encounters. As mentioned, the encounter list $\mathcal{C}_i$ of a node i is a Bloom filter. Consequently, C_i^2 is a list of Bloom filters.

D. Similarity calculation

The computation of similarity requires checking the existence of a destination identity in the set of a node's two-hop encounters. More specifically, assume that a node i has a packet to forward with dst_w^z. Node i performs a bitwise-AND operation between $dst_w^z = BF(K_{wz})$ and each encounter list stored in $\mathcal{C}_i^2$. Each time the operation results in k bits set to one, i.e. the only key in dst_w^z also exists in an encounter list, then $\text{Sim}(i, dst_w^z)$ is increased by one. Fig. 1 presents an example of A's ego network as well as the corresponding $\mathcal{C}_A$ and $\mathcal{C}_A^2$. Assume that node E sends a message to node D. As a result, the message carries the filter $dst_E^D = \text{BF}(K_{\text{ED}})$. When A is in contact with E, it performs a bitwise-AND of dst_E^D with every encounter list stored in $\mathcal{C}_A^2$. This will result in finding one match with the encounter list received by id_B^A. This means that node id_B^A is a common contact of A and the destination, thus $\text{Sim}(A, dst_E^D) = 1$.

As mentioned earlier, Bloom filters are probabilistic data structures. Thus, there is always the possibility of false positives. As a result, checking the existence of a destination identity dst_w^z in an encounter list, which is stored in $\mathcal{C}_A^2$, entails a false positive probability P'_{fp}. Keeping in mind that, in the worst case, $|\mathcal{N}|^2$ keys exist in an encounter list, then:

$$P'_{fp} \leq (1 - (1 - \frac{1}{m})^{k \cdot |\mathcal{N}|^2})^k \quad (3)$$

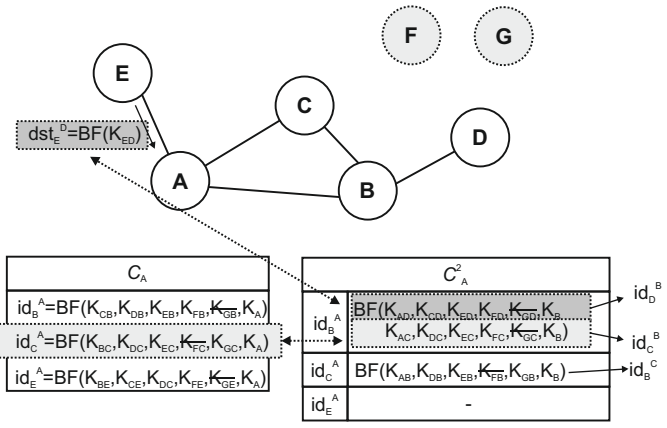


Fig. 1. Example of ego betweenness and similarity calculation

However, the actual P'_{fp} is much smaller than this limit since an encounter list usually contains much less than $|\mathcal{N}|^2$ keys (rarely a node has encountered all the other nodes). In Section VI, we will show that the impact of P'_{fp} in the performance of SimBet-BF is negligible.

E. Ego betweenness calculation

The computation of ego betweenness of a node A requires checking A's encounters in pairs of two and deciding which of them are directly connected (shortest path does not pass through A) and which are not (shortest path passes through A). In the latter case, A should also determine how many of its encounters lie between the examined pair. The check is performed by computing the bitwise-AND of each $id_j^A \in \mathcal{C}_A$ with the received encounter list from every other $id_i^A \in \mathcal{C}_A, i \neq j$. Let us go back to the example in Fig. 1. In this example, A can decide that id_C^A also exists in the encounter list received by id_B^A, therefore the nodes id_C^A and id_B^A are directly connected. This is not true for the pairs (id_C^A, id_E^A) and (id_B^A, id_E^A). Furthermore, A can check that there is no other encounter connecting those pairs. Therefore, according to (1), A's ego betweenness is 2. Notice that, by definition, for a given j, two identities id_j^z and id_j^w have at least $(|\mathcal{N}_r| - 3)$ common keys (check for example the common keys between id_C^A and id_C^B in Fig. 1). If less than $k(|\mathcal{N}_r| - 3)$ bits are set to one, then it can be concluded that id_j^z is not included in the encounters of w and thus, node z must update (increase) its betweenness value, since it lies in the shortest path connecting j and w.

The calculation of betweenness also involves a false positive probability P^*_{fp}. Since we need $k(|\mathcal{N}_r| - 3)$ bits to be set ($|\mathcal{N}_r| - 3$ keys for each id), the false positive probability is:

$$P^*_{fp} \leq (P'_{fp})^{|\mathcal{N}_r|-3} = (1 - (1 - \frac{1}{m})^{k \cdot |\mathcal{N}|^2})^{k(|\mathcal{N}_r|-3)} \quad (4)$$

which is significantly smaller than P'_{fp}, therefore its impact on routing efficiency is smaller.

F. Fine-tuning the filters and space requirements

In order to minimize the impact of false positives, the size of the Bloom filters should be appropriately chosen.

Since the highest false positive probability is the one involved in the calculation of similarity (see (3)), the size of the Bloom filters can be set to minimize this probability. If the optimal number of hash functions is used [24], then $m = -|\mathcal{N}|^2 \ln(p)/\ln(2)^2$, where p denotes the desired false positive probability. Note that this is the required size in the worst case where $|\mathcal{N}|^2$ keys exist in a node's encounter list. A more accurate expression would be to write the required size as $m = -|\mathcal{N}||\mathcal{N}_\mathcal{C}| \ln(p)/\ln(2)^2$, where $|\mathcal{N}_\mathcal{C}|$ is the maximum number of node ids stored in the encounter list. In big networks, a node is expected to meet only a small portion of network nodes, i.e., $|\mathcal{N}_\mathcal{C}| \ll |\mathcal{N}|$. Furthermore, a node's encounter list should contain only a subset of the encountered nodes in order for the routing process to be efficient [25]. This significantly alleviates the space requirements.

G. Anonymity beyond SimBet

As mentioned previously, we chose SimBet to implement the proposed privacy scheme because it represents a challenging scenario where complex computations are required for calculating both ego betweenness and similarity using the anomymized versions of C_i and C_i^2. However, observe that the proposed scheme could be applied to every algorithm that uses this type of connectivity information. Moreover, the scheme could also be used in every algorithm that simply uses some kind of utility metric (this is the case for the plethora of algorithms). For example, if an algorithm uses a utility metric that depends on the number of encounters, e.g. the PRoPHET algorithm [7], a node i could use its pool of node identities $id_i^j, j \in \mathcal{N}_i$ to distinguish different nodes and safely record the required information.

V. Privacy Analysis

The protocol is designed to withstand different kinds of privacy attacks launched by parties with different roles in the message exchanges. We discuss in the following paragraphs attacks on privacy and the defence mechanisms of the SimBet-BF protocol. In our analysis, we consider a threat model where the attacker is a legitimate node of the network that has initially received keys from the TA. The aim of the attacker is to reveal the true identity of a targeted node.

A. Message sender and destination anonymity

A major concern for network privacy relates to the role of the intermediate nodes and the access to the headers of each message. The node identities are constructed in such a way that only the final destination of a message can understand that the message is destined to it. More specifically, observe that messages stored on a node v have an address of the type $dst_i^j = \mathrm{BF}(K_{ij})$. This address cannot be related to any identity $id_v^k, k \in \mathcal{N}_v$ or $id_k^v, k \in \mathcal{N}_v$. Thus, v cannot reveal the sender node and then its identity. Only the destination node j can use $scr_j^i = \mathrm{BF}(K_{ij})$ to decide that the message was sent by node i. The maximum information, that an intermediate node can collect from the destination identities stored on the message, is that the final destination is within a 2-hop distance. This can be done through the calculation of the similarity metric. Still, it cannot even derive through which of the 1-hop nodes the final destination can be reached.

Another important security enhancement is that, when nodes A and B are in contact, A cannot directly deliver the packets destined to B and created by A (Step 2 of the encounter protocol of SimBet does not exist in the encounter protocol of SimBet-BF). This is because $dst_A^B = \mathrm{BF}(K_{\mathrm{AB}})$ is not present in id_B^A in order to block A from revealing B's identity. Instead, A forwards those packets similarly to all other packets since B will definitely claim a better SimBetUtil. In this way, A cannot distinguish whether a packet is claimed by B because B is the packet's destination or because its utility is better.

B. Node anonymity

A first line of defense relates to the fact that the node identities id_i^j are never used in protocol exchanges. Thus, an eavesdropper cannot collect such information by passively observing the `hello` broadcasts or other control messages. Moreover, in a contact between A and B, node A can use any of the source or destination identities it possesses, i.e., $src_A^i, i \in \mathcal{N}_A$ and $dst_A^i, i \in \mathcal{N}_A$, and compare it (bitwise-AND) with either id_A^B or id_B^A in order to reveal B's identity. However, all of those comparisons will end up with the same result. Thus, it is not possible for A to know that the node in contact is B.

In a previous section, we described the introduction of fake nodes identities for the benefit of security. Suppose that node A is malicious. Since it received $src_A^B = \mathrm{BF}(K_{\mathrm{BA}})$ from the TA, it can check which of the $id_A^k, k \in \mathcal{N}_A$ *does not* contain K_{BA}. By definition, this can hold only for the id_A^B. By allowing fake nodes in the network and removing real, valid keys from their identities, a malicious node cannot derive this information: it is not only id_A^B but also some fake node(s) $id_A^k, k \in \mathcal{N}_f$, which also does not contain K_{BA}. In the example of Table III, there are two identities, id_A^B and id_A^F, that do not contain K_{BA}. In general, the more keys removed, the more difficult for a malicious node to reveal another node's true identity.

C. Contact anonymity

The encounter protocol never sends the contact information in the clear but rather encoded as a bitwise-OR of the Bloom filters of the node identities. Apart from the obvious transmission economy, this approach improves the anonymity. There is no profound way that the received information can be combined with the information received from the TA so as to reveal the real identity of a node.

The computation of the similarity metric can reveal some partial information about the network formation. Indeed, the bitwise-AND operation between a destination identity dst_i^j, which is encoded in a message, and C_A^2 can reveal the existence of j in A's two-hop contacts. This is necessary for the calculation of similarity. However, this process cannot reveal A's one-hop contacts. It cannot also reveal through which contact of A, a 2-hop contact (j in our example) can be reached.

TABLE IV
PROPERTIES OF OPPORTUNISTIC TRACES

Trace Name	# Nodes	Duration (days)	Network Area
Infocom '05 [27]	41	3	conference
MIT Reality [28]	97	283	campus
Milano pmtr [27]	44	18.9	campus
Cambridge upmc [29]	52	11.4	city

Another form of attack is to combine the information from two filters received by different contacts and reveal their identity. For example, suppose that A derives through X's list that it can reach node C and derives through Y's list that it can reach node B. The worst-case scenario appears if A knows that the size of the filters (i.e., how many elements they contain) is one. In this case, $X = B$ and $Y = C$. However, the Bloom filter does not reveal the number of the elements it contains and thus, this information cannot be deduced.

VI. PERFORMANCE EVALUATION

The use of Bloom filters for the representation of a node's one-hop and two-hop encounters impacts the calculation of similarity and ego betweenness because Bloom filters are probabilistic data structures. To evaluate the impact on the routing performance, we compare SimBet and SimBet-BF through simulation. To this end, we implemented both algorithms in ONE [26] and simulated their performance under different connectivity scenarios. More specifically, we used four traces from real opportunistic networks with different characteristics. Table IV summarizes the characteristics of the used traces. For the Reality trace, due to its long duration, we simulated only the duration of one month (November). Regarding packet generation, each node produces one packet for each other node in the beginning of the simulation. This traffic pattern is often used in the literature [9].

As previously discussed, the impact of Bloom filters on the routing performance depends on the false positive probabilities P'_{fp} and P^*_{fp}. Since $P^*_{fp} < P'_{fp}$, in our experiment, we examine the performance of SimBet-BF under various levels of P'_{fp}. More specifically, let P^{max}_{fp} denote the maximum value of P'_{fp} according to (3). We examine the performance of SimBet-BF for $P^{max}_{fp} \in (0, 0.2]$. According to (3), the size of the Bloom filter m can be selected so as to result in a certain P^{max}_{fp}. Equivalently, a certain P^{max}_{fp} allows a certain trade-off between the size of the Bloom filter (m) and $|\mathcal{N}|$, i.e.

$$|\mathcal{N}| = \sqrt{\frac{1}{k}\frac{\log\{1 - \sqrt[k]{P^{max}_{fp}}\}}{\log\{1 - 1/m\}}} \quad (5)$$

Since the security level depends on the number of fake nodes $|\mathcal{N}_f| = |\mathcal{N}| - |\mathcal{N}_r|$, it is clear that, for a given P^{max}_{fp}, one can choose the security level by choosing the size of the filter m. Note that (5) refers to the worst-case scenario, i.e., when $|\mathcal{N}|^2$ identities are included in the filter. However, as previously discussed, this is rarely the case. In fact, the number of identities in a node's encounter list is bounded by $|\mathcal{N}_r| < |\mathcal{N}|$ since fake nodes do not appear as encounters.

Fig. 2a and 2b present the delivery ratio achieved by Simbet-BF under various false positive probabilities and for different traces. For comparison, we also illustrate the performance of SimBet, which does not depend on the false positive probability. It is clear that SimBet-BF's performance is slightly affected. In all cases, it delivers virtually the same percentage of messages. Interestingly enough, SimBet-BF is not affected even when the maximum false positive probability is as high as 20%. This is a confirmation that the actual false positive probability is much smaller because the encounter lists are sparsely populated. Therefore, the approach of Section IV-F can be used to setup the Bloom filters and economize on their size. Another interesting observation is that the performance of SimBet-BF is consistent in all the tested traces.

Fig. 3a and 3b illustrate the performance of both algorithms in terms of the average delay and the number of hops, respectively. Again, SimBet-BF manages to deliver messages without significantly increasing either the delay or the number of hops. In other words, it does not decrease the quality of service provided to the users nor it increases the consumption of resources in the network. Furthermore, both figures provide an indication that SimBet-BF, in most cases, makes the same routing decisions as SimBet.

VII. CONCLUSIONS

In this paper, we propose SimBet-BF, an anonymized version of the SimBet routing protocol. The protocol represents all node identities using Bloom filters. This ensures that two nodes can continue to exchange information while maintaining the privacy of their identity and their past encounters (neighborhood). SimBet-BF introduces minimal processing and communication overhead compared to the original SimBet protocol. A thorough privacy analysis reveals that the protocol can successfully defend against attacks on anonymity. Furthermore, the chosen representation is not bound to the original SimBet protocol. Rather, it can be reused to enhance alike protocols that use utility metrics based on information about encounters. We aim to explore this direction in a future work.

ACKNOWLEDGMENT

This work was partially supported by the CHOReOS project (FP7/2007-2013 under grant agreement number 257178).

A.G. Voyiatzis was supported by COMET K1, FFG - Austrian Research Promotion Agency.

REFERENCES

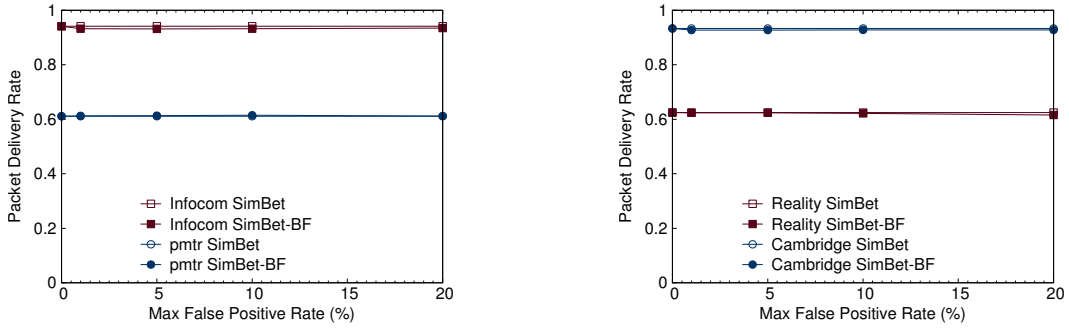
[1] A. Voyiatzis, "A survey of delay- and disruption-tolerant networking applications," *Journal of Internet Engineering*, vol. 5, no. 1, pp. 331–344, 2012.

[2] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Mask: anonymous on-demand routing in mobile ad hoc networks," *Wireless Communications, IEEE Transactions on*, vol. 5, no. 9, pp. 2376–2385, September 2006.

[3] L. Zhuang, F. Zhou, B. Y. Zhao, and A. Rowstron, "Cashmere: Resilient anonymous routing," in *Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2*, ser. NSDI'05, 2005, pp. 301–314.

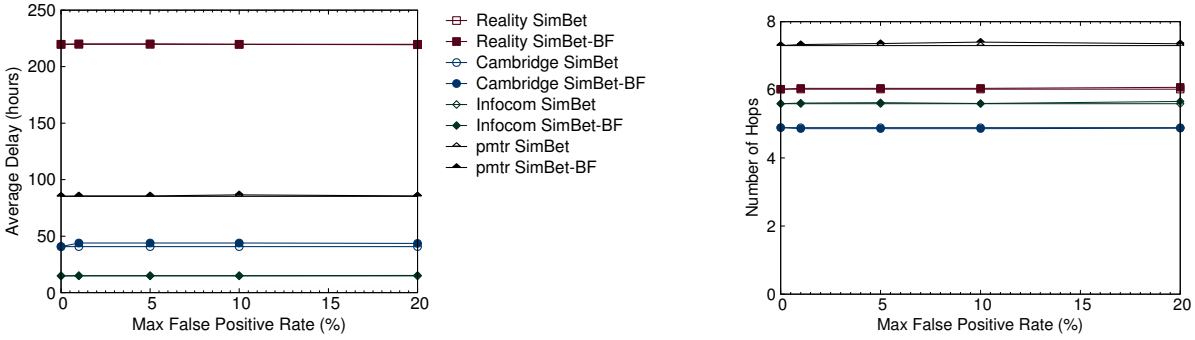
[4] A. Vahdat, D. Becker *et al.*, "Epidemic routing for partially connected ad hoc networks," Duke University, Tech. Rep., CS-200006, 2000.

[5] R. Ramanathan, R. Hansen, P. Basu, R. Rosales-Hain, and R. Krishnan, "Prioritized epidemic routing for opportunistic networks," in *Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*. ACM, 2007, pp. 62–66.



(a) (b)

Fig. 2. Delivery ratio of SimBet and SimBet-BF for different traces.



(a) (b)

Fig. 3. Performance of SimBet and SimBet-BF for different traces: (a) Average delay, and (b) Number of hops

[6] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant networking*. ACM, 2005, pp. 252–259.

[7] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE mobile computing and communications review*, vol. 7, no. 3, pp. 19–20, 2003.

[8] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 11, pp. 1576–1589, 2011.

[9] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2007, pp. 32–40.

[10] N. Papanikos and E. Papapetrou, "Coordinating replication decisions in multi-copy routing for opportunistic networks," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2014 IEEE 10th International Conference on*, Oct 2014, pp. 8–13.

[11] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Communications of the ACM*, vol. 42, no. 2, pp. 39–41, 1999.

[12] H. Choi, P. McDaniel, and T. F. La Porta, "Privacy preserving communication in manets," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON'07. 4th Annual IEEE Communications Society Conference on*. IEEE, 2007, pp. 233–242.

[13] Y. Zhu and Y. Hu, "Making peer-to-peer anonymous routing resilient to failures," in *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*. IEEE, 2007, pp. 1–10.

[14] A. Kate, G. M. Zaverucha, and U. Hengartner, "Anonymity and security in delay tolerant networks," in *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*. IEEE, 2007, pp. 504–513.

[15] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology – CRYPTO 2001*. Springer, 2001, pp. 213–229.

[16] R. Lu, X. Lin, and X. Shen, "Spring: A social-based privacy-preserving packet forwarding protocol for vehicular delay tolerant networks," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.

[17] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[18] A. Broder, M. Mitzenmacher, and A. B. I. M. Mitzenmacher, "Network applications of bloom filters: A survey," in *Internet Mathematics*, 2002, pp. 636–646.

[19] E. Papapetrou, E. Pitoura, and K. Lillis, "Speeding-up cache lookups in wireless ad-hoc routing using bloom filters," in *Personal, Indoor and Mobile Radio Communications, 2005. PIMRC 2005. IEEE 16th International Symposium on*, vol. 3, Sept 2005, pp. 1419–1423.

[20] I. Parris, G. Bigwood, and T. Henderson, "Privacy-enhanced social network routing in opportunistic networks," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*. IEEE, 2010, pp. 624–629.

[21] I. Parris and T. Henderson, "Privacy-enhanced social-network routing," *Computer Communications*, vol. 35, no. 1, pp. 62 – 74, 2012.

[22] L. Ding, B. Gu, X. Hong, and B. Dixon, "Articulation node based routing in delay tolerant networks," in *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on*, March 2009, pp. 1–6.

[23] P. V. Marsden, "Egocentric and sociocentric measures of network centrality," *Social Networks*, vol. 24, no. 4, pp. 407 – 422, 2002.

[24] S. Tarkoma, C. Rothenberg, and E. Lagerspetz, "Theory and practice of bloom filters for distributed systems," *Communications Surveys Tutorials, IEEE*, vol. 14, no. 1, pp. 131–155, First 2012.

[25] T. Hossmann, T. Spyropoulos, and F. Legendre, "Know thy neighbor: Towards optimal mapping of contacts to social graphs for dtn routing," in *INFOCOM, 2010 Proceedings IEEE*, March 2010, pp. 1–9.

[26] A. Keränen, J. Ott, and T. Kärkkäinen, "The one simulator for dtn protocol evaluation," in *Proceedings of the 2Nd International Conference on Simulation Tools and Techniques*, ser. Simutools '09. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009, pp. 55:1–55:10.

[27] "CRAWDAD data sets: Infocom 05 (v. 2006-11-14), SigComm 09 (v. 2012-07-15), unimi-pmtr (v. 2010-09-10)," Downloaded from http://crawdad.org, Mar. 2014.

[28] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Pers. and Ubiq. Comput.*, vol. 10, no. 4, pp. 255–268, 2006.

[29] J. Leguay, A. Lindgren, J. Scott, T. Friedman, and J. Crowcroft, "Opportunistic content distribution in an urban setting," in *Proc. ACM SIGCOMM Workshop Challenged Netw. (CHANTS)*, 2006, pp. 205–212.