PReFilter: An Efficient Privacy-preserving Relay Filtering Scheme for Delay Tolerant Networks

Rongxing Lu[†], Xiaodong Lin[‡], Tom Luan[†], Xiaohui Liang[†], Xu Li[§], Le Chen[†], and Xuemin (Sherman) Shen[†]
[†]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada
[‡]Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, Ontario, Canada
[§]INRIA Lille - Nord Europe, Univ Lille Nord de France, USTL, CNRS UMR 8022, LIFL, France
Email: {rxlu, hluan, x27liang, xshen}@bbcr.uwaterloo.ca; xiaodong.lin@uoit.ca; xu.li@inria.fr;l84chen@ece.uwaterloo.ca

***Abstract*—Without direct path, information delivery in sparse delay tolerant networks (DTNs) typically relies on intermittent relays, making the transmission not only unreliable but also time consuming. To make the matter even worse, the source nodes may transmit some encrypted "junk" information, similar as the spam emails in current mail systems, to the destinations; without effective control, the delivery of encrypted junk information would significantly consume the precious resource of DTN and accordingly throttle the network efficiency. To address this challenging issue, we propose PReFilter, an efficient privacy-preserving relay filter scheme to prevent the relay of encrypted junk information early in DTNs. In PReFilter, each node maintains a specific filtering policy based on its interests, and distributes this policy to a group of "friends" in the network in advance. By applying the filtering policy, the friends can filter the junk packets which are heading to the node during the relay. Note that the keywords in the filtering policy may disclose the node's interest/preference to some extent, harming the privacy of nodes, a privacy-preserving filtering policy distribution technique is introduced, which will keep the sensitive keywords secret in the filtering policy. Through detailed security analysis, we demonstrate that PReFilter can prevent strong privacy-curious adversaries from learning the filtering keywords, and discourage a weak privacy-curious friend to guess the filtering keywords from the filtering policy. In addition, with extensive simulations, we show that PReFilter is not only effective in the filtering of junk packets but also significantly improve the network performance with the dramatically reduced delivery cost due to the junk packets.**

***Keywords* – Delay tolerant networks; Relay filtering; Privacy-preserving**

I. Introduction

Delay Tolerant Networks (DTNs), such as inter-planetary communication, networking in sparsely population areas [1], and vehicular ad hoc networks [2], have been subject to extensive research efforts in recent years. Different from the traditional networks, DTNs are distinguishably characterized by high transmission delays, frequent link disruptions, and non-exist end-to-end connections. As such, the packet propagation process in DTNs is typically composed of multiple relays in a store, carry and forward fashion upon the node contacts. As the node contacts are often uncoordinated and purely unpredictable, especially in sparse DTNs, packet delivery is therefore opportunistic. To increase the delivery ratio and reduce the average delivery delay, an extensive body of research has been devoted recently to the DTN routing, and a variety of multicopy DTN routing schemes/protocols have been proposed for sparse DTNs [3]–[5].

Despite the significant progress, the challenges imposed by the security concerns [6] have, however, often been neglected in DTN routing; if the security issues are not resolved, those well-designed DTN routing schemes [3]–[5] cannot be practical for real-world deployments, not mention to exert their vantage. Aiming at security in DTN routing, there are mainly two efforts underway in research. One primarily focuses on the incentive mechanism in DTN to stimulate selfish DTN nodes to faithfully forward packets [7], [8] during the relay. Another one is to address the fundamental packet authentication and access control [9]. The fundamental packet authentication mainly utilizes the signature technique to deal with the traffic storm problem [6], [9]: if a malicious node inserts some false packets in the network, the signature verification executed by intermediate relay nodes can efficiently detect the forged signatures, drop the inserted false packets and save the scarce DTN resources. Although the fundamental packet authentication can filter false packets inserted by malicious nodes, it cannot use signature to identify those "junk" packets which are encrypted, sent by the legitimate nodes, but are of no interest to the destination nodes. Of course, the encrypted junk packets can be decrypted and discarded by the destinations immediately upon the receipt. However, before their reaching the destinations, the scarce DTN resources have already been wasted. Therefore, similar to the forged packets inserted by malicious nodes, those encrypted junk packets are equally harmful which would also lead to significant wastes of scarce DTN resources if without effective control.

Fig. 1. Replicas of an encrypted "junk" packet consuming a large amount of network storages in DTNs

Fig. 1 illustrates the case when no any treatments are made

on the junk packets. In this case, as a relay node cannot tell whether an encrypted packet is "junk" to its destination or not, it will carry a replica of the packet during the relay until either the Time-to-Live (TTL) of the packet expires or it contacts the destination and finishes relaying. Let N_s denote the packet size, k be the number of replicas in the network, and t_i be the caching time of the ith replica. The total cost spent in caching the junk packet in the network-wide relay nodes, named as network storage cost (NSC), is as $N_s \cdot \sum_{i=1}^{k} t_i$. If TTL is large in DTNs[1], the cumulative caching time of all the replicas $\sum_{i=1}^{k} t_i$ will amount to a large value, resulting in the large consumption of network resources on caching useless junk packets, especially for a large N_s. In a nutshell, to filter the junk packets in DTNs while without harming to the delivery of ordinary packets (i.e., packets useful to destinations as opposed to the junk packets) is a challenging and fundamental security issue in DTNs. However, to the best of our knowledge, this issue has not yet been actively exploited before, which motivates our research.

In this paper, aiming to address the above challenging issue, we propose an efficient Privacy-preserving Relay Filtering scheme, called PReFilter, for DTNs. In PReFilter, each vertex in DTNs has a group of intimate nodes, namely friend nodes[2], and distributes its filtering policy to its friends in advance. Based on the keywords defined in the filtering policy, the friends of a vertex can help identify and block the junk packets heading to the vertex early in a fully distributed manner during the relay. As a direct result, the scarce DTN buffer and relay resources can be save for delivering the ordinary packets. Moreover, PReFilter will not reveal any sensitive keywords of nodes in their filtering policies to the others. Therefore, using PReFilter, the user privacy, e.g., the interest/preference privacy reflected in filtering keywords can be effectively protected. To summarize, the main contributions of this paper are threefold.

- First, we address the "junk" packet issue in DTNs and propose PReFilter, an efficient privacy-preserving relay filtering scheme to remedy it. In PReFilter, based on the existing social connections, each destination node deploys certain privacy-preserving filtering policy to its friend nodes in advance. Whenever these friends encounter any junk packets in DTNs, they can filter and prevent those packets from further relays, and therefore reduce the cost incurred for the unnecessary relays.
- Second, we analyze PReFilter theoretically and unveil the impacts of TTL selection in PReFilter, and the average delay and network storage cost required for forwarding an ordinary beneficial packet. Since the junk packets would consume storage and forward resources if they are not prevented, the theoretical results show the cost of the resource due to junk packets, which accordingly confirms the importance of filtering junk packets in DTNs.
- Third, we carry out extensive simulations to examine the delivery ratio and average delay of packets, and the network storage cost due to junk packets in PReFilter. The simulation results can well validate our theoretical analysis. In addition, the extensive simulation results also demonstrate the effectiveness of PReFilter in filtering junk packets and saving resources in DTNs.

The remainder of this paper is organized as follows. In Section II, we formalize the system including the network model, node social connection model and security model, and identify our research goal. We present the detailed design of PReFilter in Section III, followed by the security analysis and performance evaluation in Section IV and Section V, respectively. Section VI reviews the related works in details and Section VII closes the paper with the conclusion.

II. Models and Design Goal

In this section, we formalize the network model, node social connection model and security model, and identify our design goal as well.

A. Network Model & Node Social Connection Model

Delay tolerant networks (DTNs) are usually characterized by the intermittent connectivity with low node density and unpredictable mobility. In this work, we consider sparse DTN network which is composed of a set of n nodes $\mathbb{N} = \{N_1, N_2, \cdots, N_n\}$ with a finite transmission range R moving in a sparse area, as shown in Fig. 2. All nodes follow the same random-based mobility model (i.e. random way point, random direction) with equal velocity v. As reported in [5], [10], using this model the pairwise node inter-contact time can be considered to be exponentially distributed, and the contacts between any pairs of nodes form a homogeneous Poisson process with the same contact rate λ, where λ is dependent on the mobility model, velocity and transmission range of nodes. In such a DTN, large-size packets are stored and carried by moving nodes for a long time, and forwarded upon the contact with the next relay node. Using this "store, carry and forward" approach, the packets can finally be delivered to their destinations in an opportunistic manner.

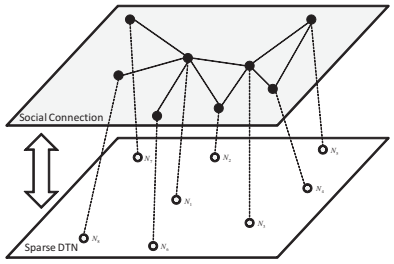


Fig. 2. Network model & node social connection model under consideration

Although the physical contacts of DTN nodes are opportunistic in a spare environment, we consider that these exist certain social connections among nodes in advance, as shown

[1]The TTL value of a replica at each relay node is necessarily large enough to guarantee that the replica does not expire and are removed before the contacts of relays.

[2]The friend nodes of a vertex may refer to the nodes of certain social connections to the vertex or close to the vertex with frequent contacts. We assume that the group of friend nodes of a vertex is predefined for each vertex.

in Fig. 2; the nodes which have social connections to a vertex are referred to as friends to the vertex. In PReFilter, a node $N_i \in \mathbb{N}$ can distribute its filtering policy to its friends, even to the friends of its friends. Apparently, the more social connections and friends a node has, the more widely its filtering policy can be distributed in the network. In our model, we use the number of direct friends of a node N_i to measure its social degree, denoted as $\deg(N_i)$, and we consider a simple case in which node N_i only distributes its filtering policy to its direct friends. Once a friend node of N_i applies the filtering policy of N_i and detects a junk packet to N_i in DTN, the node will early drop the junk packet to avoid the potential waste of network resources.

B. Security Model

Unlike previously reported security research in DTNs [6]–[9], our work does not focus on either malicious or selfish nodes to degrade the performance of DTN. Instead, we consider how to use the distributed filtering policy deployed at friend nodes to early detect and filter the encrypted junk packets from the legitimate sources to reduce the transmission costs of the junk packets. Meanwhile, PReFilter protects a node's interest/preference privacy-preserving as well. Specifically, we consider "weak privacy-curious friend" (WPCF) and "strong privacy-curious adversary" (SPCA) in our security model, where the WPCF is privacy-curious to i) infer what filtering keywords embedded in its friend's filtering policy, and ii) identify what packet's contents are being relayed to its friend. The privacy curiosity of WPCF is assumed weak, i.e., if identifying the filtering keywords in filtering policy needs certain costs, the WPCF will not take any action to satisfy his curiosity. SPCA has the same privacy curiosity as WPCF. However, SPCA's privacy curiosity is much stronger, i.e., the costs will not stop his privacy curiosity. Although a WPCF is privacy-curious, it is still faithful to its friend, and will not collude with SPCA. Therefore, the collusion between WPCF and SPAC is beyond the scope of this paper.

C. Design Goal

Our design goal is to develop an efficient privacy-preserving relay filtering mechanism to filter those large-size junk packets as early as possible to avoid the significant waste of the DTN resources. Specifically, the following two desirable goals should be achieved.

- *Efficient Utilization of DTN Network Resource.* The network resource in terms of transmission opportunity in DTNs is scarce in nature due to the rare contacts of nodes. Therefore, the proposed relay filtering mechanism, due to the distributed filtering policy deployed at friend nodes, should help to efficiently utilize the network resources. In other words, those encrypted junk packets heading to their destinations should be early filtered and not relayed in DTNs.
- *Privacy-preserving Filtering Policy Distribution.* A filtering policy is utilized for filtering junk packets. However, the distribution of filter-policy could disclose a node's interest/preference privacy. Therefore, to prevent a SPCA from learning the filtering keywords, and discourage a WPCF to guess the filtering keywords from the filtering policy, the devised filtering policy should be privacy-preserving.

III. Proposed PReFilter Scheme

In this section, we propose our PReFilter scheme, which consists of three parts: system initialization, privacy-preserving filtering policy distribution, and packet forwarding with relay filtering, followed by the TTL parameter selection and complexity analysis on the filtering policy. Before plunging into the details, we first review the bilinear pairing technique [11], [12], which serves as the basis of the proposed PReFilter scheme.

A. Bilinear Pairings

Let $\mathbb{G}$, $\mathbb{G}_T$ be two cyclic groups with the same prime order q. Suppose $\mathbb{G}$ and $\mathbb{G}_T$ are equipped with a pairing, i.e., a non-degenerated and efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ such that $e(aP_1, bP_2) = e(P_1, P_2)^{ab} \in \mathbb{G}_T$ for all $a, b \in \mathbb{Z}_q^*$ and any $P_1, P_2 \in \mathbb{G}$. In group $\mathbb{G}$, the Collusion Attack Algorithm with k traitors (k-CAA) Problem is hard, i.e., given $(P, Q = xP, h_1, h_2, \cdots, h_k \in \mathbb{Z}_q^*, \frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \cdots, \frac{1}{h_k+x}P)$, there is no polynomial-time algorithm which can compute $\frac{1}{h^*+x}P$ for some $h^* \notin \{h_1, h_2, \cdots, h_k\}$ with non-negligible probability. We refer to [11], [12] for a more comprehensive description of pairing techniques, and complexity assumptions.

Definition 1: A bilinear parameter generator $\mathcal{G}en$ is a probabilistic algorithm that takes a security parameter κ as input, and outputs a 5-tuple $(q, P, \mathbb{G}, \mathbb{G}_T, e)$, where q is a κ-bit prime number, $\mathbb{G}, \mathbb{G}_T$ are two groups with order q, $P \in \mathbb{G}$ is a generator, and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a non-degenerated and efficiently computable bilinear map.

B. Description of PReFilter

1) System Initialization Phase: For a single-authority DTN system under consideration, we assume a trusted authority (TA) will bootstrap the whole system. Specifically, given the security parameter κ, TA first generates the bilinear parameters $(q, P, \mathbb{G}, \mathbb{G}_T, e)$ by running $\mathcal{G}en(\kappa)$, and chooses a secure symmetric encryption algorithm $\mathcal{E}nc()$, i.e., AES, and two collision-resistant cryptographic hash functions $\mathcal{H}_i : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ with $i = 0, 1$. In addition,TA also chooses a random number $s \in \mathbb{Z}_q^*$ as the master key, and computes $P_{pub} = sP$. Finally, TA keeps the master key s secretly, and publishes the system parameters $params = (q, P, P_{pub}, \mathbb{G}, \mathbb{G}_T, e, \mathcal{H}_0, \mathcal{H}_1, \mathcal{E}nc())$.

For each node $N_i \in \mathbb{N}$, when it registers itself in the system, the following steps will be run between TA and N_i:

- N_i first chooses a random number $x_i \in \mathbb{Z}_q^*$ as private key, computes the corresponding public key $Y_i = x_i P$, and submits (N_i, Y_i) to TA for registration.
- After receiving (N_i, Y_i), TA first checks the eligibility of node N_i, calculates $cert_i = \frac{1}{s+\mathcal{H}_0(N_i||Y_i)}P$ to bind N_i and Y_i, and return $cert_i$ to N_i;

- Since anyone can check $e(cert_i, P_{pub}+\mathcal{H}_0(N_i||Y_i)P) \stackrel{?}{=} e(P,P)$ for confirming the binding of (N_i, Y_i), node N_i can use the key materials $(x_i, Y_i, cert_i)$ to achieve secure packet forwarding in the system later.

2) Filtering Policy Distribution Phase: For each node $N_i \in \mathbb{N}$, to avoid the "junk" packets consuming the scarce DTN resources in time period $time$, it first chooses n_i keywords $\mathbb{K}_i = [\mathbf{k}_{i1}, \mathbf{k}_{i2}, \cdots, \mathbf{k}_{in_i}]$ that it has interests in, and makes the filtering policy $FP_i = [rule_1, rule_2, \cdots, rule_{n_i}]$, where each $rule_\alpha = \frac{1}{x_i+\mathcal{H}_0(\mathbf{k}_{i\alpha})}P$. In order to make the filtering mechanism efficiently, N_i distributes the filtering policy FP_i to its friends in advance, as shown in Fig. 3. For each friend N_j with public key $Y_j = x_jP$, N_i generates the filtering policy $FP_i^j = [rule_1^j, rule_2^j, \cdots, rule_{n_i}^j]$ by invoking Algorithm 1, and distributes FP_i^j to N_j. Later, when N_j moves in DTN, it can use FP_i^j to filter those packets "junk" to N_i, i.e., the packet does not contain any keyword listed in $\mathbb{K}_i$.

Algorithm 1 Filtering Policy Establishment

procedure FILTERINGPOLICYESTABLISHMENT
  on input of n_i keywords $\mathbb{K}_i = [\mathbf{k}_{i1}, \mathbf{k}_{i2}, \cdots, \mathbf{k}_{in_i}]$ and N_j's public key Y_j
  for each keyword $\mathbf{k}_{i\alpha} \in \mathbb{K}_i = [\mathbf{k}_{i1}, \mathbf{k}_{i2}, \cdots, \mathbf{k}_{in_i}]$ **do**
    choose a random number $r_\alpha \in \mathbb{Z}_q^*$
    calculate $R_\alpha = \frac{\mathcal{H}_0(r_\alpha Y_j||time)}{x_i+\mathcal{H}_0(\mathbf{k}_{i\alpha})}P$, $S_\alpha = r_\alpha P$
    set $rule_\alpha^j =< R_\alpha, S_\alpha >$
  end for
  return $FP_i^j = [rule_1^j, rule_2^j, \cdots, rule_{n_i}^j]$
end procedure

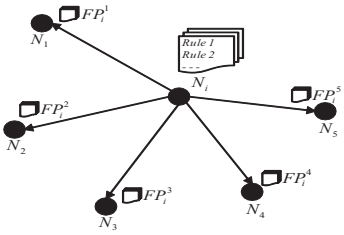


Fig. 3. Filtering policy distribution to friends for the filtering efficiency

3) Packet Forwarding with Relay Filtering Phase: Assume a static source node S wants to securely send a large-size packet M labeled with a keyword $\mathbf{k}_s$ to a mobile node $N_d \in \mathbb{N}$ with public key Y_d in DTN. S will perform the following steps:

Step 1: S first chooses a random number $r \in \mathbb{Z}_q^*$, and computes $C = (C_1, C_2, C_3)$, where

$$\begin{cases} C_1 = r(Y_d + \mathcal{H}_0(\mathbf{k}_s)P), \\ C_2 = e(P,P)^r, \\ C_3 = \mathcal{E}nc_{sk}(M), \text{where } sk = \mathcal{H}_1(e(P,Y_d)^r) \end{cases} . \quad (1)$$

Step 2: When a mobile node $N_i \in \mathbb{N}$ passes by S at time T_0, S first chooses a proper Time-to-live (TTL) label TTL, and utilizes the short signature algorithm [12] to make a signature $sign$ on $N_d||C_1||C_2||C_3||TTL$. Then, S formats the packet as shown in Fig. 4 and sends the packet to N_i[3].

[3]Note that, in PReFilter, we consider a legitimate node S will not send the same packet to other nodes, once it has already sent the packet to N_i.

Algorithm 2 Relay Filtering Examination

procedure RELAYFILTERINGEXAMINATION
  On input of $C_1||C_2$ and a filtering policy *FP*
  set token = 0
  if *FP* is $FP_d = [rule_1, rule_2, \cdots, rule_{n_d}]$ held by N_d **then**
    for each $rule_\alpha \in FP_d = [rule_1, rule_2, \cdots, rule_{n_d}]$ **do**
      if $C_2 == e(C_1, rule_\alpha)$ **then**
        set token = 1
        break
      end if
    end for
  else if *FP* is $FP_d^j = [rule_1^j, rule_2^j, \cdots, rule_{n_d}^j]$ held by a friend N_j of N_d **then**
    for each $rule_\alpha^j =< R_\alpha, S_\alpha > \in FP_d^j$ **do**
      if $C_2 == e(C_1, \frac{1}{\mathcal{H}_0(x_jS_\alpha||||time)}R_\alpha)$ **then**
        set token = 1
        break
      end if
    end for
  end if
  return token
  /* correctness: since $x_jS_\alpha = x_jr_\alpha P = r_\alpha Y_j$

$$\begin{aligned} & e\left(C_1, \tfrac{1}{\mathcal{H}_0(x_jS_\alpha||time)}R_\alpha\right) \\ = \; & e\left(C_1, \tfrac{1}{\mathcal{H}_0(x_jS_\alpha||time)} \cdot \tfrac{\mathcal{H}_0(r_lY_j||time)}{x_d+\mathcal{H}_0(\mathbf{k}_{d\alpha})}P\right) \\ = \; & e\left(C_1, \tfrac{1}{x_d+\mathcal{H}_0(\mathbf{k}_{d\alpha})}P\right) = e(C_1, rule_\alpha) \\ = \; & e\left(r(Y_d+\mathcal{H}_0(\mathbf{k}_s)P), \tfrac{1}{x_d+\mathcal{H}_0(\mathbf{k}_{d\alpha})}P\right) \\ = \; & e\left(r(x_d+\mathcal{H}_0(\mathbf{k}_s))P, \tfrac{1}{x_d+\mathcal{H}_0(\mathbf{k}_{d\alpha})}P\right) \quad \underline{\text{when } \mathbf{k}_s = \mathbf{k}_{d\alpha}} \\ = \; & e(P,P)^r = C_2 \quad */ \end{aligned}$$

end procedure

Destination	Keyword & Payload	TTL	Authenticator
N_d	$C_1 \| C_2 \| C_3$	*TTL*	*Sign*

Fig. 4. Packet format in transmission

Upon receiving the packet, N_i first verifies the authenticity of the packet by checking $sign$. Then, if N_i happens to be the destination N_d, the destination invokes the Algorithm 2 with the filtering policy FP_d to check whether the packet is "junk" or not. If the returned value is 1, the destination uses its private key x_d to recover the message M from $C_3 = \mathcal{E}nc_{sk}(M)$ with $sk = \mathcal{H}_1(C_2{}^{x_d}) = \mathcal{H}_1(e(P,Y_d)^r)$. Otherwise, the destination drops the packet. If N_i is one of friends of the destination N_d, N_i first invokes the Algorithm 2 with FP_d as well. If the returned value is 1, N_i will help to forward the packet. Otherwise, N_i drops the packet, and informs the source node S that the packet is out of the interests of N_d so that S will not send the packet again. If N_i is neither the destination N_d nor a friend of N_d, N_i just helps to forward the packet.

Since the packets are forwarded in a sparse DTN, the node contacts are not quite frequent. Therefore, in order to improve delivery ratio and reduce the delivery delay, multi-copy forwarding with relay filtering policy is considered. Especially, for any node $N_i \in \mathbb{N}/\{N_d\}$ who carries the packet, the following Algorithm 3 will be invoked when N_i contacts another node $N_j \in \mathbb{N}$.

Algorithm 3 Multicopy Forwarding with Relay Filtering

procedure MULTICOPYFORWARDINGWITHRELAYFILTERING
  a mobile node N_i carries a packet $\mathbf{pck} = (C_1||C_2||C_3)$ in DTN
  while packet's TTL has not expired **do**
    once N_i contacts another node $N_j \in \mathbb{N}$, they run the packet forwarding procedure
    if N_i is a friend of N_d with filtering policy FP_d^i **then**
      if N_j is the destination node N_d **then**
        N_i forwards the packet to N_d and removes its own replica; upon receiving the packet, if N_d received the packet's replica before, N_d just ignores the packet; otherwise, N_d recovers the message M from $C_3 = \mathcal{E}nc_{sk}(M)$ with $sk = \mathcal{H}_1({C_2}^{x_d})$.
      else
        N_i just forwards the packet to N_j
      end if
    else /* N_i is not a friend of N_d */
      if N_j is the destination node N_d **then**
        N_i forwards the packet to N_d and removes its own replica; upon receiving the packet, if N_d received the packet's replica before, N_d just ignores the packet; otherwise, N_d invokes the Algorithm 2 with $C_1||C_2$ and FP_d. If the returned value is 1, N_d uses private key x_d to recover the message M from $C_3 = \mathcal{E}nc_{sk}(M)$ with $sk = \mathcal{H}_1({C_2}^{x_d})$). Otherwise, N_d drops the packet.
      else if N_j is a friend of N_d with the filtering policy FP_d **then**
        N_i forwards the packet to N_j
        if N_j has already held the packet's replica **then**
          N_j ignores the forwarding
        else if N_j once dropped the packet **then**
          N_j ignores the forwarding, and informs N_i to drop the packet as well. The reason is either the packet is a "junk" packet or the destination N_d has already received it.
        else if N_j has not gotten the packet yet **then**
          N_j invokes the Algorithm 2 with $C_1||C_2$ and FP_d^j. If the returned value is 1, N_j stores the packet. Otherwise, both N_j and N_i drop the packet.
        end if
      else if N_j is not a friend of N_d **then**
        N_i forwards the packet to N_j. If N_j has already held the packet, N_j ignores the forwarding. If N_j once dropped the packet, N_j ignores the forwarding and informs N_i to drop the packet as well. Otherwise, if N_j has not gotten the packet yet, N_j will hold the packet.
      end if
    end if
  end while
  N_i drops the packet /* TTL has expired */
end procedure

C. Analysis on TTL Parameter Selection

For any multicopy forwarding protocol in DTNs, the TTL parameter selection is crucial for the tradeoff between Quality of Delivery (QoD) and network resource consumptions. If the TTL is set too short, the destination may have no chance to receive the packet before the TTL expires. On the other hand, if the TTL is overly long, there will exist too many replicas stored in the network and consuming the DTN resources. To analyze the TTL in the proposed PReFilter scheme, we first evaluate the average delay for the destination N_d receiving an ordinary beneficial packet when there are k replicas being propagated in the network.

Theorem 1: The average delays of the $(k+1)$-th replica generation, for $k+1 < n$, and the n-th replica generation of an ordinary packet in the proposed PReFilter scheme are $\Theta\left(\frac{1}{\lambda n}\ln\frac{k(n-1)}{n-k-1}\right)$ and $\Theta\left(\frac{2}{\lambda n}\ln(n-1)\right)$, respectively.

Proof: For the proposed PReFilter scheme, if the source node S directly contacts the destination N_d at time T_0, there is no replica propagating in the network; otherwise, a replica is generated and the packet opportunistically reaches the destination N_d in a multicopy forwarding pattern. Let T_k be the start time that there exist k replicas in the network, where $k \le n-1$, then $t_k = T_{k+1} - T_k$ denotes the time period in which the $(k+1)$-th replica is generated. Since all pairwise contacts follow a homogeneous Poisson process with contact rate λ in our network model, by the memoryless property, the rate for the first contact between a node carrying a replica and a node without a replica is $k(n-k)\lambda$. As such, the average delay t_k is $\frac{1}{k(n-k)\lambda}$. With the observation, the average delay for the $(k+1)$-th replica is generated from T_0 to T_{k+1} can be expressed as:

$$
\begin{aligned}
&E[T_{k+1} - T_0] = \sum_{i=1}^{k} E[T_{i+1} - T_i] = \sum_{i=1}^{k} E[t_i] \\
=&\frac{1}{(n-1)\lambda} + \frac{1}{2(n-2)\lambda} + \cdots + \frac{1}{k(n-k)\lambda} \\
=&\frac{1}{\lambda n}\left(\frac{n}{n-1} + \frac{n}{2(n-2)} + \cdots + \frac{n}{k(n-k)}\right) \\
=&\frac{1}{\lambda n}\left(1 + \frac{1}{n-1} + \frac{1}{2} + \frac{1}{(n-2)} + \cdots + \frac{1}{k} + \frac{1}{(n-k)}\right) \\
=&\begin{cases} \frac{1}{\lambda n}\left(\sum_{i=1}^{k}\frac{1}{i} + \sum_{j=1}^{n-1}\frac{1}{j} - \sum_{l=1}^{n-k-1}\frac{1}{l}\right), & k < n-1; \\ \frac{2}{\lambda n}\sum_{i=1}^{n-1}\frac{1}{i}, & k = n-1. \end{cases} \\
=&\begin{cases} \frac{(H_k + H_{n-1} - H_{n-k-1})}{\lambda n} \approx \frac{1}{\lambda n}\ln\frac{k(n-1)}{n-k-1}, & k < n-1; \\ \frac{2}{\lambda n}H_{n-1} \approx \frac{2}{\lambda n}\ln(n-1), & k = n-1. \end{cases}
\end{aligned}
\tag{2}
$$

where $H_a = \ln a + \gamma + O(\frac{1}{a})$, $a \in \{k, n-1, n-k-1\}$, is the a-th harmonic number with the Euler's constant $\gamma = 0.5772\cdots$. Therefore, the average delay for the $(k+1)$-th replica generation is $\Theta\left(\frac{1}{\lambda n}\ln\frac{k(n-1)}{n-k-1}\right)$, and the average delay for the n-th replica generation is $\Theta\left(\frac{2}{\lambda n}\ln(n-1)\right)$. In other words, the average delay to disseminate an ordinary packet in the whole network requires $\Theta\left(\frac{2}{\lambda n}\ln(n-1)\right)$. ■

Theorem 2: In the proposed PReFilter scheme, the probability that an ordinary beneficial packet reaches its destination within time period $|T_{k+1} - T_0|$ is $\frac{k+1}{n}$.

Proof: Let B_k denote the event that the destination N_d receives the packet when there are already k replicas being propagated in the network. Then, in the proposed PReFilter scheme, the probability $\Pr(B_0) = \frac{1}{n}$ is obvious, and the probability $\Pr(B_k)$, for $k \ge 1$, can be written as

$$
\begin{aligned}
\Pr(B_k) &= \frac{1}{n-k} \cdot \prod_{i=0}^{k-1}\left(1 - \frac{1}{n-i}\right) \\
&= \frac{n-1}{n} \cdot \frac{n-2}{n-1} \cdot \cdots \cdot \frac{n-k}{n-k+1} \cdot \frac{1}{n-k} = \frac{1}{n}
\end{aligned}
\tag{3}
$$

Consider all events $B_1, \cdots, B_k$, we have

$$
\Pr\left(\bigcup_{i=0}^{k} B_i\right) = \sum_{i=0}^{k}\Pr(B_i) = \sum_{i=0}^{k}\frac{1}{n} = \frac{k+1}{n} \tag{4}
$$

Therefore, the probability that an ordinary beneficial packet reaches its destination within the time period $|T_{k+1} - T_0|$ is $\frac{k+1}{n}$. ■

From the above two theorems, when TTL is set as $\Theta\left(\frac{1}{\lambda n}\ln\frac{k(n-1)}{n-k-1}\right)$, the expected QoD shows that $(k+1)/n$ percent of ordinary beneficial packets can reach their destinations before the TTL expires.

As for the network resource consumptions, we consider all replicas of an ordinary beneficial packet will be removed when the TTL expires. Then, the following theorem shows the average network storage cost (NSC) required for forwarding a packet within the time period $|T_{k+1} - T_0|$.

Theorem 3: In the proposed PReFilter scheme, the average NSC required for forwarding a packet within the time period $|T_{k+1}-T_0|$ is $\Theta(\frac{N_s}{\lambda}\ln\frac{n-1}{n-k-1})$ for $k < n-1$, and $\Theta(\frac{N_s}{\lambda}\ln(n-1))$ for $k = n-1$, where N_s is the packet size.

Proof: According to the Theorem 1, the average caching time for the first replica from T_0 to T_{k+1} in the network is $E[T_{k+1} - T_0]$. Since the second replica is generated at time T_1, the average caching time for the second replica can be expressed as $E[T_{k+1} - T_1]$. By repeating the processes for the i-th replica, where $i \le k$, we have the average NSC as

$$
\begin{aligned}
&E(\text{NSC}) = \sum_{i=1}^{k} E[T_{k+1} - T_{i-1}] \cdot N_s \\
=&N_s \cdot \left(\frac{1}{(n-1)\lambda} + \frac{1}{2(n-2)\lambda} + \cdots + \frac{1}{k(n-k)\lambda} + \right. \\
&\frac{1}{2(n-2)\lambda} + \frac{1}{3(n-3)\lambda} + \cdots + \frac{1}{k(n-k)\lambda} + \cdots + \\
&\left. \cdots + \frac{1}{(k-1)(n-k+1)\lambda} + \frac{1}{k(n-k)\lambda} + \frac{1}{k(n-k)\lambda} \right) \\
=&\frac{N_s}{\lambda}\left(\frac{1}{n-1} + \frac{1}{n-2} + \cdots + \frac{1}{n-k} \right) \\
=&\begin{cases} \frac{N_s}{\lambda}\left(\sum_{i=1}^{n-1}\frac{1}{i} - \sum_{j=1}^{n-k-1}\frac{1}{j}\right), & k < n-1; \\ \frac{N_s}{\lambda}\sum_{i=1}^{n-1}\frac{1}{i}, & k = n-1. \end{cases} \\
=&\begin{cases} \frac{N_s}{\lambda}\left(H_{n-1} - H_{n-k-1}\right) \approx \frac{N_s}{\lambda}\ln\frac{n-1}{n-k-1}, & k < n-1; \\ \frac{N_s}{\lambda}H_{n-1} \approx \frac{N_s}{\lambda}\ln(n-1), & k = n-1. \end{cases}
\end{aligned}
\tag{5}
$$

Therefore, the average NSC required for forwarding a packet within the time period $|T_{k+1} - T_0|$ is $\Theta(\frac{N_s}{\lambda}\ln\frac{n-1}{n-k-1})$ for $k < n-1$, and $\Theta(\frac{N_s}{\lambda}\ln(n-1))$ for $k = n-1$. Note that, in the proposed PReFilter scheme, when a node carrying a replica happens to know the destination has already received the packet, it will drop the replica before the TTL expires. In this case, the $E(\text{NSC})$ will reduce slightly. ■

The above analysis demonstrates how the TTL selection affects the QoD and NSC. Obviously, in order to improve the QoD of an ordinary beneficial packet, the TTL should be set large. Then, we can take more NSC to ensure the packet can reach the destination with a high probability. However, for a "junk" packet, when TTL is set as $\Theta\left(\frac{1}{\lambda n}\ln\frac{k(n-1)}{n-k-1}\right)$, it will also waste $\Theta(\frac{N_s}{\lambda}\ln\frac{n-1}{n-k-1})$ network storage resource, and the longer TTL will waste even more scarce resources in DTN. Therefore, how to preserve the network storage resources is crucial for the performance improvement in DTN. In the proposed PReFilter scheme, the distributed filtering policies deployed at friend nodes can efficiently filter those "junk" packets as early as possible. Thus, the network storage resources can be saved. In Section V, we will conduct extensive simulations to further evaluate the effectiveness of the proposed PReFilter scheme.

D. Complexity Analysis on Filtering Policy

In our complexity analysis, the bilinear pairing e from $\mathbb{G}$ to $\mathbb{G}_T$ is implemented with a Tate pairing [11], where each element in $\mathbb{G}$ is 512-bit in length, and the order q is a 160-bit prime. According to [13], this kind of Tate pairing can achieve the same security level as 1024-bit RSA, and the computation cost of Tate pairing is around 8.5 ms on a Pentium III 1 GHz machine. For a filtering policy $FP_d^j = [rule_1^j, rule_2^j, \cdots, rule_{n_d}^j]$ deployed at a friend node N_j of the destination N_d, since each rule $rule_\alpha^j =< R_\alpha, S_\alpha >$ is formed by two elements in $\mathbb{G}$, so FP_d^j takes around $n_d \times 1024$ bits. According to the Algorithm 2, the relay filtering examination requires $n_d \times 0.85$ ms. When $n_d = 1000$, the size of FP_d^j is about 1 M, and the computation cost of filtering is only 0.85 second. Compared with the end-to-end delay in DTN, the computation delay can be negligible.

IV. Security Discussion

In this section, we discuss the security properties of the proposed PReFilter scheme. In specific, following the security model discussed earlier, our discussion will focus on how the proposed PReFilter scheme can achieve the privacy-preserving filtering policy distribution, and how the proposed PReFilter scheme can also achieve other basic security requirements.

A. The proposed PReFilter scheme can achieve the privacy preserving filtering policy distribution.

To achieve the privacy-preserving filtering policy distribution, the prerequisite is that the keywords in filtering rules cannot appear in plaintext. In the proposed PReFilter scheme, when a node N_i distributes a filtering rule on keyword $\mathbf{k}_{i\alpha}$ to its friend N_j in advance, the rule $rule_\alpha^j =< \frac{\mathcal{H}_0(r_\alpha Y_j||time)}{x_i + \mathcal{H}_0(\mathbf{k}_{i\alpha})}P, r_\alpha P >$ will not disclose the keyword $\mathbf{k}_{i\alpha}$ directly to N_j. If the friend node N_j is really interested in knowing the filtering keyword, it should take much more time to identify it. For example, if the keyword space $\mathcal{KS}$ is not too large, N_j can exhaustively try each keyword $\mathbf{k}_x \in \mathcal{KS}$ by checking whether or not

$$
\begin{aligned}
&e(Y_i + \mathcal{H}_0(\mathbf{k}_x)P, \frac{1}{\mathcal{H}_0(x_j r_\alpha P||time)} \cdot \frac{\mathcal{H}_0(r_\alpha Y_j||time)}{x_i + \mathcal{H}_0(\mathbf{k}_{i\alpha})}P) \\
=&e(P, P)
\end{aligned}
$$

If it does hold, N_j is guessing the correct keyword $\mathbf{k}_{i\alpha} \in \mathcal{KS}$. In addition, since the friend-dependent random element $\mathcal{H}_0(r_\alpha Y_j||time)$ is embedded in the filtering rule $rule_\alpha^j$, N_j can only guess the keyword from the filtering rule deployed at itself in time period $time$, but has no idea on the filtering

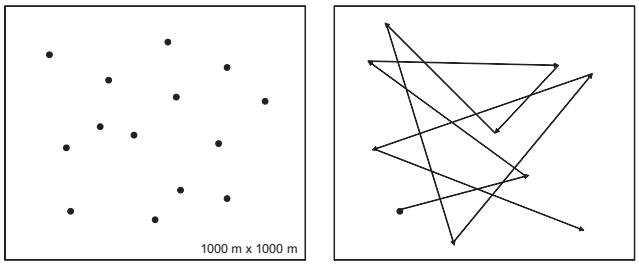
keywords deployed at other friends, once different keywords are distributed at different friends. Therefore, the long-time keyword guessing process and its limitation will discourage a WPCF to exhaustively guess its friend filtering keywords. On the other hand, even if a SPCA has eavesdropped the filtering rule $rule_{\alpha}^{j} =< \frac{\mathcal{H}_0(r_\alpha Y_j||time)}{x_i+\mathcal{H}_0(\mathbf{k}_{i\alpha})} P, r_\alpha P >$, it still cannot identify the keyword $\mathbf{k}_{i\alpha}$ by exhaustively trying all keywords in $\mathcal{KS}$, because it cannot eliminate the item $\mathcal{H}_0(r_\alpha Y_j||time)$ from $rule_{\alpha}^{j}$. By summarizing the above two aspects, the proposed PReFilter scheme can prevent a SPCA from learning the keywords in filtering policy, and discourage a WPCF to guess the keywords. As a result, it can achieve the privacy-preserving filtering policy distribution.

B. The proposed PReFilter scheme can also satisfy other basic security requirements.

In the proposed PReFilter scheme, when the source node S generates the packet pck shown in Fig. 4, the signature $Sign$ can provide the source authentication and integrity protection, because the employed signature algorithm is provably-secure under k-CAA problem in the random oracle model [12]. If an active adversary launches some forgery and/or modification attacks, the forged and/or modified packets can be detected easily. In addition, since the message M is encrypted as $C_1||C_2||C_3$ in the packet, only the destination N_d can use its private key x_d to recover the message M from $C_3 = \mathcal{E}nc_{sk}(M)$ with $sk = \mathcal{H}_1(C_2{}^{x_d})$. As a result, the proposed PReFilter scheme can also achieve the confidentiality, while it efficiently filters "junk" packets in DTN.

V. Performance Evaluation

In this section, we evaluate the performance of the proposed PReFilter scheme using a custom simulator built in Java. The simulator implements the network layer and assumes each DTN node contact has sufficient capacity to exchange large-size packet. The performance metrics used for the evaluation are delivery ratio (DR), average delay (AD), and average network storage cost (NSC), where the DR is defined as the ratio of the ordinary beneficial packets successfully delivered to their destinations with respect to total packets within a given time period, the AD is defined as the average time between when a packet is generated and when it is successfully delivered to the destination, and NSC is defined as the average network storage consumptions for a packet within a time period between when the packet is generated and when the packet's TTL expires.



(a) Simulation area (b) Random waypoint model

Fig. 5. Simulation area and mobility model under consideration

A. Simulation Setup

To simulate a DTN scenario, $n = \{20, 30, 40\}$ DTN nodes with transmission radius of $tr = \{25, 35\}$ m are first deployed in a region of 1000 m $\times$ 1000 m, as shown in Fig. 5(a). Then, with the same velocity $v = \{1, 2\}$ m/s, each node independently moves in the region by following the random waypoint model, as shown in Fig. 5(b). In the simulation, we assume that all nodes have enough storage to carry the packet they receive, and the bandwidth is also high enough to allow the exchange of packets upon nodes contact. In view of capabilities of today's communication technology, these assumptions are reasonable.

In order to show the effectiveness of the proposed PReFilter scheme in DTN, we evaluate how the factors n, v, tr affect the DR, AD, and NSC for ordinary beneficial packet forwarding, and also examine different number of relay filters (NRF) detecting the "junk" packets to reduce the NSC. For each simulation run, we assume that a random static source node contacts a mobile DTN node N_i at the beginning of the simulation, and the source generates a packet for a randomly chosen destination N_d among n DTN nodes, where the packet's TTL is set equal to the simulation duration. After the mobile DTN node gets the packet, the packet will be opportunistically forwarded to the destination N_d in a multicopy with relay filtering way. The detailed parameter settings are summarized in Table I.

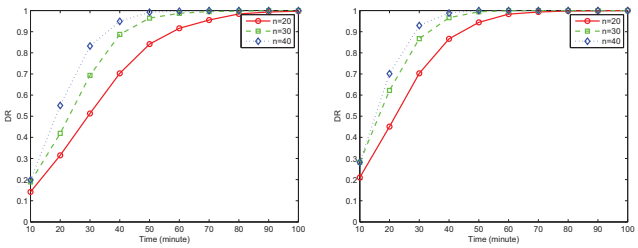
TABLE I
Simulation Settings

Parameter	Setting
Simulation area	$1,000$ m $\times$ $1,000$ m
Simulation duration	100 minutes
DTN node	
number	$n = \{20, 30, 40\}$
velocity, transmission radius	$v = \{1, 2\}$ m/s, $tr = \{25, 35\}$ m
mobility model	random waypoint model
Packet size, generation time, TTL	N_s, 0, 100 minutes
The number of relay filters	NRF $= \{2, 4, 6, 8\}$

In the following, we conduct the simulations with different parameter settings. For each case, we run the simulation for 100 minutes, and the average performance results over 10000 runs are reported.

B. Simulation Results

Delivery ratio: Fig. 6 depicts the delivery ratio in forwarding ordinary beneficial packets with different number of nodes n, velocity v and transmission radius tr. From the figure, we can see the delivery ratio increases as the time increases in all cases, as expected. When the node density increases, many packet replicas will be generated and carried in the network, and then the destination can receive the packet early with high probability. Therefore, Fig. 6 also shows the larger the number of nodes n, the faster the delivery ratio reaches closely to 1. Further observing the four subfigures, both the velocity v and the transmission radius tr can positively affect the delivery ratio, the reason is that the two factors v, tr enable a node to contact other nodes more frequently.

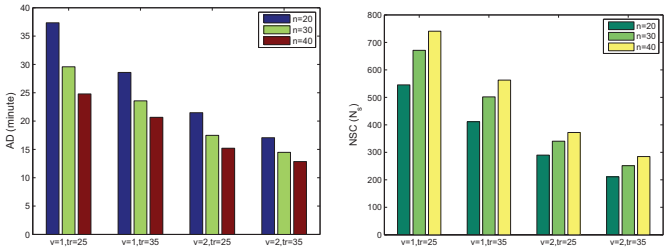


(a) $v = 1$ m/s, $tr = 25$ m (b) $v = 1$ m/s, $tr = 35$ m

(c) $v = 2$ m/s, $tr = 25$ m (d) $v = 2$ m/s, $tr = 35$ m

Fig. 6. Delivery ratio in forwarding ordinary beneficial packets with different n, v and tr

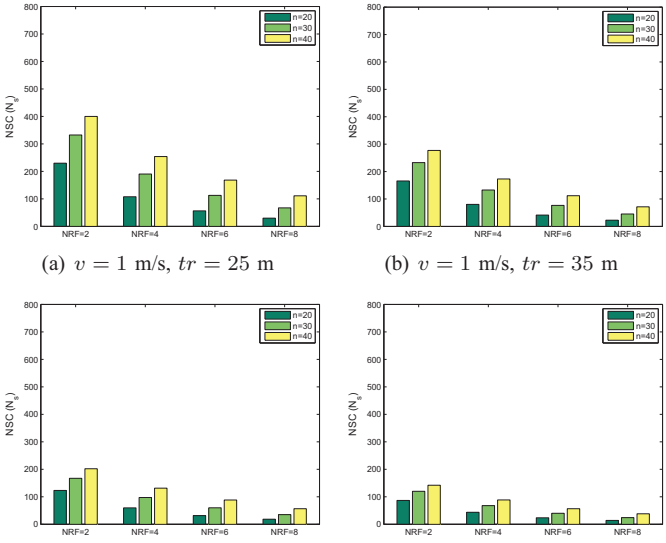
Average delay and network storage cost: Fig. 7 shows the corresponding average delay and network storage cost in forwarding an ordinary beneficial packet. From the Fig. 7 (a), we can see the increases of the number of nodes n, the velocity v and the transmission radius tr will decrease the average delay, which corresponds to the theoretical analysis result $\Theta\left(\frac{1}{\lambda n}\ln\frac{k(n-1)}{n-k-1}\right)$. Since the contact rate $\lambda \propto \alpha \cdot v \cdot tr$ for some mobility model dependent constant α [10], $AD \propto \frac{1}{v}$, $AD \propto \frac{1}{tr}$. In addition, since $\frac{1}{n}\ln\frac{k(n-1)}{n-k-1}$ is a decreasing function with respect to n, for $k < n-1$, we have $AD \propto \frac{1}{n}\ln\frac{k(n-1)}{n-k-1}$ as well. From the Fig. 7 (b), we can see the increases of the velocity v and the transmission radius tr can decrease the NSC, while the increase of n will increase the NSC, which also corresponds to the theoretical analysis result $\Theta(\frac{N_s}{\lambda}\ln\frac{n-1}{n-k-1})$, i.e., $NSC \propto \frac{1}{v}$, $NSC \propto \frac{1}{tr}$, and $NSC \propto \ln\frac{n-1}{n-k-1}$. Note that, $\ln\frac{n-1}{n-k-1}$ here is an increasing function with respect to n for $k < n-1$.



(a) Average delay (b) Network storage cost

Fig. 7. Average delay and network storage cost in forwarding an ordinary beneficial packet with different n, v, and tr

Network storage cost under PReFilter: Fig. 8 shows the average network storage wasted in forwarding a "junk" packet with different number of relay filters NRF under different n, v, tr. From the figure, we can observe that, due to the relay filtering, the network storage wasted in forwarding a "junk" packet is obviously lower than that consumed in forwarding an ordinary beneficial packet indicated in Fig. 7(b). The more the relay filters, the lower the network storage are wasted, especially in the scenario with large node velocity $v = 2$ m/s and transmission radius $tr = 35$ m. To accurately show why PReFilter can remove "junk" packets to save network storage, we further plot the average replicas distribution of a "junk" packet in Fig. 9. From the figure, we can make the following observations. When NRF is small and node density is large, more nodes can get a packet replica before the relay filter gets a replica at the early stage. Therefore, the larger the number of nodes n, the more replicas will be generated in the network. However, once the relay filter gets a replica of "junk" packet, the replica will be immediately filtered. Since the node density is high, other replicas can be quickly dropped as well. However, when NRF is large, the replica of a "junk" packet can meet a relay filter early, and thus the replicas will not increase significantly. In addition, the larger v and tr can also accelerate the relay filters to take actions to filter the replicas of a "junk" packet in the network. As a result, the wasted network storage can be greatly reduced, and this demonstrates the effectiveness of PReFilter.



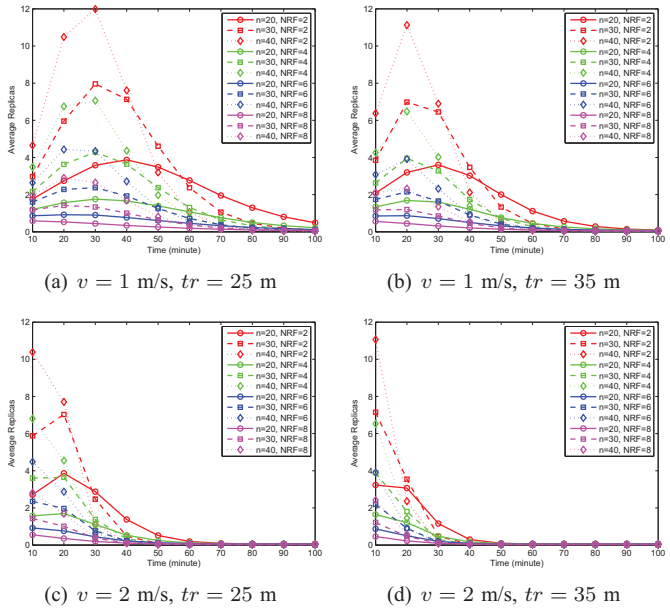
(a) $v = 1$ m/s, $tr = 25$ m (b) $v = 1$ m/s, $tr = 35$ m

(c) $v = 2$ m/s, $tr = 25$ m (d) $v = 2$ m/s, $tr = 35$ m

Fig. 8. Average network storage cost in forwarding a "junk" packet under PReFilter with different NRF, n, v, and tr

VI. Related Works

Recently, there have appeared several research works on decentralized collaborative filtering in wireless networks [14]–[17], which are closely related to the proposed PReFilter scheme. In [14], Maccari et al. propose a distributed firewalling with Bloom filter for multi-hop mesh network. In the scheme, each node adopts a bloom filter to represent all packets accepted by the node, and then distributes the bloom filter to all nodes in the network. When a node is ready to forward a packet, it will first check the packet with all bloom filters it has received from others. If the packet really exists in one of bloom filters, the node will forward the packet. Otherwise, the packet will be dropped. With this



(a) $v = 1$ m/s, $tr = 25$ m (b) $v = 1$ m/s, $tr = 35$ m

(c) $v = 2$ m/s, $tr = 25$ m (d) $v = 2$ m/s, $tr = 35$ m

Fig. 9. Average replicas distribution of a "junk" packet under PReFilter with different NRF, n, v, and tr

decentralized packet filtering mechanism, all unwanted packets can be dropped early in the network, and as a result, the network resource can be saved. Although the bloom filter can compact the data structure, the efficiency in the scheme is still a challenging issue once the amount of accepted packets is huge in the mesh networks. In [15], Fantacci et al. further discuss the efficient packet filtering in wireless ad hoc networks, highlight the advantages of the use of filtering technique based on Bloom filter, and suggest using more efficient d-left hash-based Bloom filter to address the efficiency. In [16], Alicherry et al. present a distributed firewall architecture for mobile ad hoc networks, which is based on the concept of network capabilities, and can protect both end-host resources and network bandwidth from denial of service attacks. Most recently, Taghizadeh et al. [17] propose a collaborative firewalling scheme for mobile networks, where they introduce two performance metrics, namely Packet Discarding Ratio (PDR) and Forwarding Cost Ratio (FCR), to address the effectiveness of distributed firewalling in the network. In addition, a heuristic algorithm, namely the Split Replacement Policy, is presented to maximize PDR and minimize FCR. Extensive simulations show that by distributing only 1% of filtering rules, about 42% of the unwanted traffic is discarded before it reaches the destination, thus it significantly saves the network resources.

Although our proposed PReFilter scheme addresses the same issue to filter "junk" packets early to save the network resources, in contrast to the above works, our research focuses are different: i) we address the privacy-preserving filtering policy distribution, which can keep the sensitive filtering keywords privacy in filtering rules; and ii) we propose our relay filtering mechanism in more challenging delay tolerant networks, and theoretically demonstrate that multicopy forwarding with relay filtering, due to filtering "junk" packets early, can avoid the significant waste of network resources.

VII. Conclusions

In this paper, we have proposed a privacy-preserving relay filtering (PReFilter) scheme for DTNs, which mainly exploits how to achieve privacy-preserving filtering policy distribution to friend nodes, and opportunistically use them as relay filters to filter "junk" packets as early as possible to avoid the significant waste of scarce network resources. Detailed security analysis shows that the proposed PReFilter scheme can not only prevent a strong privacy-curious adversary from learning the filtering keywords in filtering policy, but also discourage a weak privacy-curious friend to guess the filtering keywords. In addition, through extensive performance evaluation, we have also demonstrated the proposed PReFilter scheme, due to the opportunistic relay filtering mechanism, can save many network storage resources in DTNs. In our future work, we will consider the heterogenous mobility of DTN nodes, and distribute the filtering policy at high-social friend nodes to make the relay filtering mechanism more efficient.

References

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. of SIGCOMM 2003*, Germany, 2003, pp. 27–34.

[2] R. Lu, X. Lin, and X. Shen, "Spring: A social-based privacy-preserving packet forwarding protocol for vehicular delay tolerant networks," in *Proc. of IEEE INFOCOM'10*, 2010, pp. 632–640.

[3] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19–20, 2003.

[4] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proc. of WDTN '05*, 2005, pp. 252–259.

[5] U. Lee, S. Y. Oh, K.-W. Lee, and M. Gerla, "Relaycast: Scalable multicast routing in delay tolerant networks," in *Proc. of ICNP'08*, 2008.

[6] N. Asokan, K. Kostiainen, P. G. J. Ott, and C. Luo, "Towards securing disruption-tolerant networking," Nokia Research, Tech. Rep. NRC-TR-2007-007.

[7] S. Upendra, H. H. Song, L. Qiu, and Y. Zhang, "Incentive-aware routing in dtns," in *Proc. of ICNP'08*, 2008, pp. 238–247.

[8] R. Lu, X. Lin, H. Zhu, X. Shen, and B. R. Preiss, "Pi: a practical incentive protocol for delay tolerant networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, pp. 1483–1493, 2010.

[9] H. Zhu, X. Lin, R. Lu, X. Shen, D. Xing, and Z. Cao, "An opportunistic batch bundle authentication scheme for energy constrained dtns," in *Proc. IEEE INFOCOM'10*, 2010, pp. 605–613.

[10] R. Groenevelt, P. Nain, and G. Koole, "The message delay in mobile ad hoc networks," *Perform. Eval.*, vol. 62, pp. 210–228, 2005. [Online]. Available: http://dx.doi.org/10.1016/j.peva.2005.07.018

[11] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," in *Proc. of CRYPTO'01*, 2001.

[12] F. Zhang, R. Safavi-Naini, and W. Susilo, "An efficient signature scheme from bilinear pairings and its applications," in *Proc. of PKC'04*, 2004, pp. 277–290.

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

[14] L. Maccari, R. Fantacci, P. N. Ayuso, and R. M. Gasca, "Mesh network firewalling with bloom filters," in *Proc. of ICC'07*, 2007, pp. 1546–1551.

[15] Leonardo Maccari, Pablo Neira Ayuso, Romano Fantacci, Rafael M. Gasca, "Efficient packet filtering in wireless ad-hoc networks," *IEEE Communications Magazine*, pp. 104–110, 2008.

[16] M. Alicherry, A. D. Keromytis, and A. Stavrou, "Deny-by-default distributed security policy enforcement in mobile ad hoc networks," in *SecureComm*, 2009, pp. 41–50.

[17] M. Taghizadeh, A. R. Khakpour, A. X. Liu, and S. Biswas, "Collaborative firewalling in wireless networks," in *Proc. of INFOCOM'11*, 2011, pp. 46–50.