accepted to appear in Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS 2019) - Workshop on Network Meets Intelligent Computations (NMIC), July, Dallas, TX

Integrating In-Network Computing for Secure and Efficient Cascaded Delivery in DTNs

Eyuphan Bulut* and Murat Yuksel†
*Dept. of Comp. Science, Virginia Commonwealth University, Richmond, VA 23284
†Dept. of Elec. and Comp. Engineering, University of Central Florida, Orlando, FL 32816
Email: ebulut@vcu.edu, murat.yuksel@ucf.edu

***Abstract*—Delay tolerant networks can facilitate communication in the aftermath of disasters and emergency situations. However, the routing of messages between the nodes in such sparsely connected networks could be challenging. There have been many algorithms proposed to increase the delivery likelihood of messages while staying in the limitations of such challenged environments. In almost all of these studies, the routing problem has been considered between a source and destination pair. However, the communication between different source-destination pairs may be correlated and the delivery of one message may trigger another message routing process (e.g., a response back or separate message to another node). In this paper, we study such a *cascaded delivery* process in delay tolerant networks, in which there is a chain of source-destination pairs that are connected in terms of their message generation. We utilize a multi-copy based routing scheme and propose integrating in-network computing at relay nodes in order to achieve an efficient routing scheme with increased delivery ratio and reduced delay without increasing the number of message forwardings. Moreover, to address the potential privacy issues, we also utilize homomorphic encryption based computations. We evaluate the proposed scheme via simulations and show that it can improve the routing performance without releasing the content of intermediate messages to unintended destinations.**

***Index Terms*—Delay tolerant networks, routing protocol, cascaded delivery, efficiency, homomorphic encryption.**

I. Introduction

Delay Tolerant Networks (DTNs) were originally proposed for building a communication infrastructure for interplanetary Internet. Yet, within the last two decades (starting mostly with the seminal talk by Kevin Fall [1]), the academic and industrial wireless community has recognized the DTNs' potential and performed extensive research on the topic. Moreover, Delay Tolerant Networking (DTN)[1] has grown as new network applications such as mobile social networks and vehicular DTNs, where intermittent connectivity is considered the norm unlike the traditional always-connected networking paradigms. DTNs consist of sparsely connected nodes which can only communicate intermittently. This makes the network suffer from frequent partitions and creates a challenged environment for the delivery of a message from a source to a destination node. Due to the uncertainty in nodal mobility, DTN routing algorithms usually adopt an opportunistic and multiple copy based routing strategy. That is, when nodes come to the transmission range of each other, they forward a copy of their messages to the other nodes to increase the delivery likelihood. While a complete flooding-based approach (i.e., copying to every node met) can potentially yield the optimal delivery, it causes a huge overhead and most of the time due to the limited buffer space at nodes, the actual delivery ratio becomes too low. Thus, approaches that limit the copying [2]–[4] have been adopted in several DTN routing designs. Moreover, copying decision is either determined randomly or probabilistically [5] depending on the characteristics of the interactions between the nodes. For example, in 'social' DTNs, where the nodes are carried by people (e.g., mobile social networks), the social relations between people have been analyzed to design efficient copy-based routing algorithms [6]–[9].

[1]We will use the acronym DTN to express 'Delay Tolerant Networking' or 'Delay Tolerant Network' interchangeably.

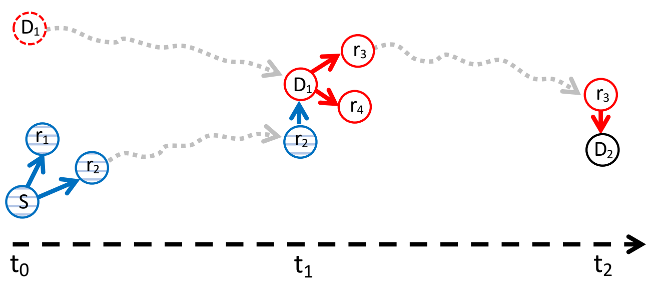

Fig. 1: An example of cascaded delivery with two destinations. Once a message, m_1, from source node S to the first destination D_1 is delivered, the routing of a next message, m_2, from D_1 to the next destination D_2 starts.

Despite the variety of the studies [10] for routing of messages in DTNs, the problem most of the time has been considered between a source node and a destination node, or with multiple destinations in case of multicasting [11]. In other words, communication needs between different pairs of nodes in the network have usually been handled independently from each other and the same routing algorithm has been utilized for each source-destination pair. However, the nodes in a DTN may communicate sequentially (e.g., query-response style) and the communication needs between different pairs might depend on each other. We call this process as *cascaded delivery* and illustrate an example scenario with two destinations in Fig. 1. Source node S first needs to deliver its message, m_1,

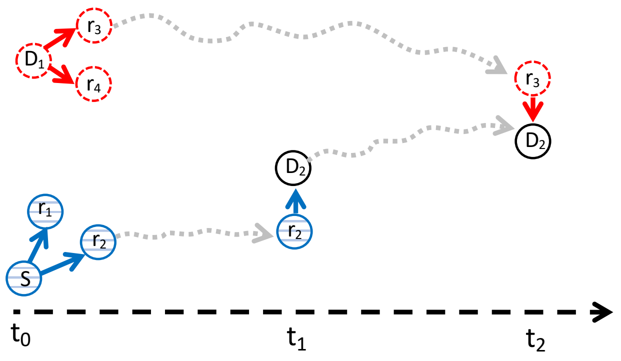

Fig. 2: Joint delivery: The source node S and the intermediate destination D_1 in the cascaded route send their messages and necessary information directly to the final destination (D_2) for computation of final message content at D_2.

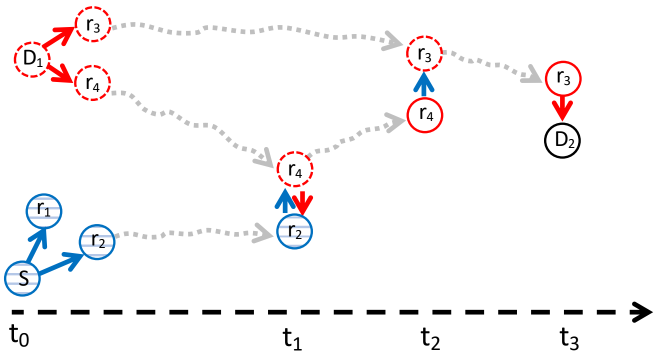

Fig. 3: In-network-computed (INC) delivery: The source S and intermediate destination D_1 distribute a copy of their messages to the relay nodes in the network and the computation of final message content happens at relay nodes. Final destination, D_2, then receives one of the computed final message content from any of these relays.

to the first destination (D_1). It uses two relay nodes (r_1, r_2) through this process and delivery happens via r_2 at time t_1. Once D_1 receives this message, it generates a new message, m_2, and aims a delivery through new relays r_3 and r_4 (which can be r_1 or r_2). Delivery happens by one of the carriers of the message copy, i.e., r_3, at time t_2.

A cascaded delivery process, as in Fig. 1, can happen naturally in different real life scenarios. Consider a DTN scenario in the aftermath of a disaster, where a victim first needs to send a message to the rescue team leader. Then, this leader needs to process it, merge with its own knowledge and send it to say a local rescue officer which is the closest to the victim and can provide the help with minimum delay. Such a cascaded delivery may also be applicable for data collection or information update purposes. A message created by a source node may need to arrive at different nodes to collect their updates and come back to the source node. This may need to happen in a sequential manner in order to be cost efficient as well as to benefit intermediate destinations from already collected updates. This makes the communication between different source-destination pairs correlated and can benefit from new routing strategies exploiting this correlation, since independent handling of the delivery of each message may result in waste of resources and low delivery ratios at the final destination.

One way to speed up the delivery at the final destination could be starting the routing process simultaneously at all source nodes by targeting the same final destination and making the necessary computations at that node[2]. We call this approach *joint delivery* and illustrate it in Fig. 2. Both S and D_1 start the routing process at time t_0 with their own relays and once the final destination, D_2, receives both of these messages, it can generate the final message by performing necessary computations on them. Note that the actual message required to be delivered to D_2 is m_2 and it would normally be computed by the first destination, D_1, with the delivery of m_1 created by source S. Thus, D_1 will only be able to deliver the additional information required to perform the computation (this is indicated by dashed circle) to D_2. In Fig. 2, the delivery of m_1 to D_2 happens earlier than the delivery of required additional information from D_1. However, the other order is also possible, and in either case the computation is possible only after both messages are delivered to D_2. While this joint delivery process can speed up the delivery, it comes with other challenges. First of all, the cascaded delivery process has not been optimized in terms of the routing cost and the delivery of each message is handled independently. The originating nodes (S, D_1) for each message still distribute the same number of copies of their message to the relays without benefiting from each other. Moreover, the information from all originating nodes is unnecessarily provided to the final destination, which may also create a privacy concern by the intermediate destination nodes. *Note that we want to make the content of each message known to its specific destination only (e.g., m_1 by D_1, m_2 by D_2).* If this was not the concern, network coding based solutions (e.g., [12], [13]) could be used to deliver each individual content from all originating nodes to the destination.

To this end, we propose to integrate in-network computing at relay nodes which will meet with each other earlier than their meeting with destination nodes. We call this process *In-network computed (INC)* delivery and illustrate it in Fig.3. Both of the message originators, S and D_1, start the routing at time t_0 as in the case of joint delivery but let their relays interact with each other and perform necessary computations earlier than it is performed at the final destination in the joint delivery case. For example, r_4 carrying a copy of m_2 interacts with r_2 which has a copy of m_1 at time t_1 and a message exchange occurs between each other. This enables

[2]Note that this will require D_1 know when S has started the routing of m_1. This could be achieved easily in a pre-scheduled delivery (e.g., at every hour) or a long range radio could be used for just notification.

them to compute the actual message content that has to be delivered to the final destination. Similarly, r_4 interacts with r_3 at time t_2 and shares m_1 with it as it has already the additional information needed from D_1. While this in-network interaction and computing of relay nodes facilitate the cascaded delivery process, it comes with some challenges. As relay nodes exchange messages (mutually or one directional), the cost of the routing algorithm (i.e., number of forwardings) increases. One way to mitigate that problem is to let the message originating nodes use smaller number of relay nodes, leaving room for additional message exchanges between relays without increasing the overall delivery cost. On the other hand, this smaller relay usage may degrade the expected delivery performance. This trade-off then rises the question *"Is it possible to tune up the algorithm's relay counts to achieve an increase in the delivery rate at the final destination without increasing the cost?"* In this paper, we mainly aim to find an answer to this question. Additionally, it is possible that the cascaded delivery process might include sensitive data and each source-destination pair may not want their message content known by other nodes in the network. As the proposed INC-delivery process may let the relay nodes learn other message contents, we propose to exploit a homomorphic-based computation during this routing process.

The rest of the paper is structured as follows. We discuss the related work in Section II. In Section III, we describe the details of the proposed solution. In Section IV, we provide our initial simulation results and finally, we conclude the paper and outline future work in Section V.

II. Related Work

In the literature of DTN routing algorithms, only a few of the studies have considered sequential or cascaded delivery process, but for different situations. For example, in [14], sequential packet delivery is considered for the routing of multiple packets generated at the same source headed to the same destination. It is assumed that the nodes in the network has a buffer space of one message and epidemic routing is used for delivery. Thus, when an intermediate node receives the next generated packet, it drops the previous one, reducing the delivery probability. It has been shown that network coding can improve the delivery probability in such a sequential packet delivery scenario. However, in this scenario, there is still one source and one destination. In another work [15], sequential delivery concept is also used for different chunks of a single message from a single source to a single destination as well.

There is a large set of studies [12], [13], [16]–[18] that use network coding to optimize the process of delivery of multiple messages generated at different source nodes to the same destination node. The messages are combined at the relay nodes using different coding schemes (e.g., linear coding [17], erasure coding [13], fountain codes [18]) and the destination could derive all the individual messages after collecting sufficient amount of these coded messages. These solutions will not help here as we want the content of each specific message known by only its own destination.

In this paper, we study a different cascaded delivery scenario, that happens in a chain of source destination pairs with the ultimate goal of delivering a final message to a final destination but the generation of this message depends on the previous messages delivered to previous destination nodes. The final destination node is also only interested in the final message content and should not know the content of previous messages used to derive the final message. To the best of our knowledge, such a privacy-preserving cascaded delivery scenario has not been studied before in a holistic manner even though it can naturally appear in practice as exampled earlier.

III. Proposed Approach

The main goal of this study is to integrate in-network computing for a cascaded delivery task in a chain of source destination pairs in delay tolerant challenged environments. To this end, we first model the problem for a multi-copy based routing algorithm and elaborate on the proposed design.

A. System Model

We assume a DTN with N nodes. For the routing of messages, we assume that a simple multi-copy routing algorithm such as Spray and Wait [2] is used. There is a source node that sends the first message and there are k different sequential destinations for the delivery task, $\langle S, D_1, D_2 \dots D_k\rangle$. The ith destination is denoted by D_i and the message that has to be delivered to D_i is denoted by m_i. Each destination D_i uses a function $\mathcal{F}_i$ to generate the next message m_{i+1} that needs to be delivered to the next destination D_{i+1} using its own information s_i and the message received from previous destination (or the source node for D_1). That is:

$$m_{i+1} = \mathcal{F}_i(m_i, s_i)\ \forall i \in [1, k-1]. \quad (1)$$

For simplicity, we assume that $\mathcal{F}_i(m_i, s_i)$ is in the form $\alpha m_i + \beta s_i$, which covers quite a good range of possible operations. For example, for a cascaded delivery aiming aggregation of data collected from other nodes in a sequence, $\alpha = 1$, $\beta = 1$.

The delivery of a message m_i between its corresponding source and destination node pair (i.e., $\langle S, D_1\rangle$ for $i = 1$ and $\langle D_i, D_{i+1}\rangle \forall i \in [1, k-1]$) can be modeled using the intermeeting time relations between the DTN nodes. We assume that the intermeeting time is exponentially distributed with intensity λ, which is the inverse of the expected intermeeting time. This is a common assumption made for modeling delivery in DTNs [2], [3], [5] and analysis of real DTN traces supports it.

Let X_k denote the random variable showing the delivery likelihood (at the final destination) in a cascaded delivery process with k destinations. The CDF of this delivery probability by time t, $F_{X_k}(x = t)$, can be calculated by:

$$F_{X_k}(x = t) = \int_{t_1=0}^{t} \int_{t_2=0}^{t-t_1} \dots \int_{t_k=0}^{t-\sum_{i=0}^{k-1} t_i} \lambda^k e^{-\lambda \sum_{i=0}^{k} t_i}$$

Here, each t_i represents the delivery time for message m_i, which has a probability density function (pdf) of $\lambda e^{-\lambda t_i}$ for

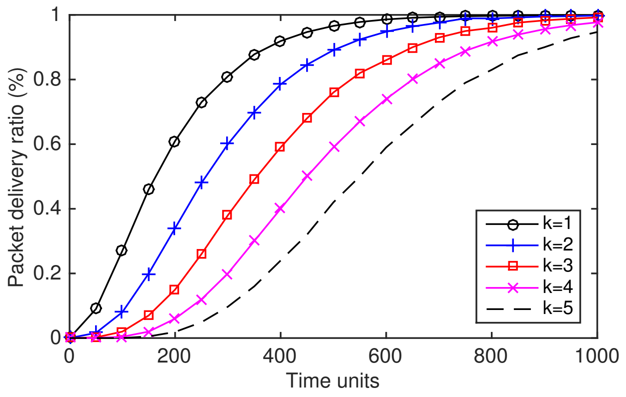

Fig. 4: Packet delivery ratios in cascaded (sequential) delivery with different number of destinations.

a delivery at time t_i. This set of integrals then gives us the following:

$$F_{X_k}(x=t) = 1 - e^{-\lambda t}\left(\sum_{i=0}^{k-1}\frac{(\lambda t)^i}{i!}\right)$$

For example, with $k=2$, this formula becomes $1-e^{-\lambda t}(1+\lambda t)$. Note that as $k\to\infty$, the last part in parenthesis becomes the Taylor series expansion of $e^{\lambda t}$, which makes $F_{X_k}\to 0$. In Fig. 4, we show the comparison of $F(X_k)$ for different k values. We also verified these with simulations using the settings that will be described later in Section IV.

On the other hand, if the routing process have started from each originating node simultaneously (i.e., joint delivery), the final expected message could be performed at the final destination. Let Y_k denote the random variable showing the delivery likelihood of all messages at the final destination D_k in a joint delivery process with k destinations (and also k corresponding originating nodes). The CDF of this delivery probability by time t, $F_{Y_k}(y=t)$, can be calculated by:

$$\begin{aligned} F_{Y_k}(y=t) &= \int_{t_1=0}^{t}\int_{t_2=0}^{t}\cdots\int_{t_k=0}^{t}\lambda^k e^{-\lambda\sum_{i=0}^{k}t_i} \\ &= \left(1-e^{-\lambda t}\right)^k \end{aligned}$$

In Fig. 5, we show the comparison of F_{X_k} and F_{Y_k} at a fixed time unit ($t=300$) up to $k=8$ destinations. These results are also verified with simulations. As the graph shows, the gap between two delivery methods increases with more destinations. While the joint delivery process is better, it comes with the cost of exposing the content of the each individual message to the destination, which may not be desired in practice. Thus, the goal of the proposed scheme is to obtain a delivery ratio as close as possible to the delivery ratio of joint delivery process while preventing the unintended destinations obtain the content of other messages.

B. Secure and Efficient Cascaded Delivery

The goal of the proposed cascaded delivery process is to let the destination receive the final message without knowing the contents of the other previous messages in the cascade. To this

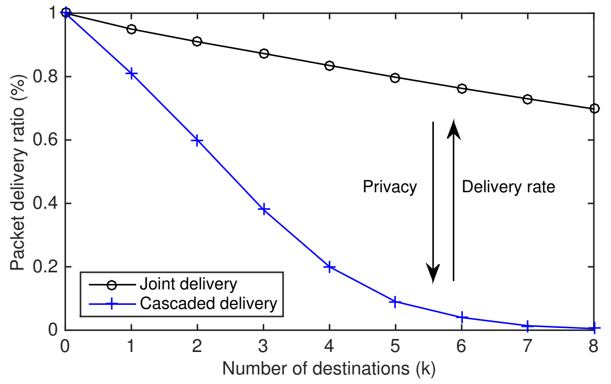

Fig. 5: Impact of destination count in the packet delivery ratios of cascaded (sequential) and joint delivery at 300 time units.

end, we want to benefit from simultaneous start of delivery process as in the aforementioned joint delivery process, but encrypt each message such that it will not be decryptable by untargeted destinations. On the other hand, the final destination should be able to use these encrypted messages to obtain the final message in plaintext, without knowing the content of each individual previous message.

We propose to use homomorphic encryption (HE) which allows computation (e.g., addition, multiplication) on ciphertexts such that when the generated encrypted result is decrypted, it matches the result of the operations as if they had been performed on the plaintext. This decryptability property enables the relay nodes to do some in-network computations earlier than they would normally be performed at their destinations and without knowing the actual contents of the messages. There are two main approaches for homomorphic encryption methods: Partially Homomorphic Encryption (PHE) and Fully Homomorphic Encryption (FHE). While the former can only support one of these operations (i.e., addition or multiplication) on the ciphertext, the latter supports both addition and multiplication. However, FHE is much slower than PHE and it could take very long to decrypt a message in a real application [19]. Therefore, we use one of the simple and efficient PHE methods available, called Palliers cryptosystem [20]. In Paillier's system when the ciphertexts of two different plaintexts are multiplied, the ciphertext of the addition of two actual plaintexts could be obtained. Moreover, with exponentiation of a ciphertext with another plaintext, the ciphertext of the multiplication of the plaintexts could be obtained. In other words, the following equations could be achieved:

$$\mathbb{E}(a).\mathbb{E}(b) = \mathbb{E}(a+b) \tag{2}$$

$$\mathbb{E}(a)^b = \mathbb{E}(ab) \tag{3}$$

where $\mathbb{E}(a)$ stands for ciphertext of a.

In Paillier's system, the encryption key is defined with a tuple (n,g) and the decryption is defined with another one (ν,γ). These are computed based on two large prime numbers p and q with same bit length as follows:

$$n = pq,\ \ g = (n+1),\ \ \nu = (p-1)(q-1)$$
$$\gamma = (\nu \bmod n^2)^{-1} \bmod n.$$

The encryption ($\mathbb{E}(a)$) and decryption ($\mathbb{D}(a)$) operations are then computed as:

$$\mathbb{E}(a) = g^a r^n \pmod{n^2}$$
$$\mathbb{D}(a) = G((\mathbb{E}(a))^\nu \bmod n^2).\gamma (\bmod\ n)$$

where r is a randomly selected integer $\in (Z)_n$ and $G(u) = (u-1)/n$. Note that these encryption and decryption computations satisfy both (2) and (3).

In the proposed scheme, each originating node encrypts their own message using the final destination's public encryption keys (which are assumed to be known by the nodes in the network) and gives a copy of this ciphertext to the relay nodes. The relay nodes, that meet with other relay nodes carrying the ciphertexts of other messages, exchange the missing ciphertexts with each other and synchronize themselves in terms of stored ciphertexts. Once a relay node has collected all the necessary ciphertexts to compute the final message and meets with the final destination, it first computes the final message and forwards it to the final destination. For example, for a cascaded delivery with two destinations, a relay node having $\mathbb{E}(m_1)$ and $\mathbb{E}(s_1)$, performs the following homomorphic operations to get $\mathbb{E}(m_2)$.

$$\mathbb{E}(m_2) = \mathbb{E}(\alpha m_1 + \beta s_1) = (\mathbb{E}(m_1))^\alpha \,.\, (\mathbb{E}(s_1))^\beta$$

The final destination then decrypts the ciphertext and obtains the raw data. This process does not allow relay nodes see the content of any other node's specific information (e.g., s_i). Similarly, the destination can only obtain what it needs from the relay nodes.

A relay node waits until all the necessary messages needed to compute the final message. For example, in a cascaded delivery with $k = 3$, if a relay with $\mathbb{E}(m_1)$ meets with another relay with $\mathbb{E}(s_2)$, it stores $\mathbb{E}(s_2)$ and waits for meeting with another relay that can provide $\mathbb{E}(s_1)$, then performs the necessary computations to get $\mathbb{E}(m_3)$, which is the message for the final destination. If the relay node with a subset of other messages meets with the final destination, it does not provide any message to it as the final computation cannot be performed due to missing messages. In fact, in some cases (e.g., with $\alpha = 1$ and $\beta = 1$), relays with a subset of messages not in the required sequence could provide a ciphertext of these messages and let the destination compute the rest. For example, in a cascaded delivery with $k = 4$, if one relay provides $\mathbb{E}(m_1+s_2)$ and the other relay provides $\mathbb{E}(s_1 + s_3)$, D_4 could obtain $\mathbb{E}(m_4)$ by multiplication. However, this should not cause any information leakage to the final destination node. Thus, this process has to be performed very carefully and relays should not provide a ciphertext which has common messages as this can cause decryption of some individual messages at the final destination.

In the proposed INC delivery process, the selection of relay count is also critical as we want to keep the cost of routing (i.e., number of message forwardings) similar or less than the cost of sequential or joint delivery. Let L denote the number of relay nodes (including the originating node) that will be used for the routing of each message between each source destination pair, the cost of sequential or joint delivery for delivered packets converges to Lk (which includes forwarding of copies to the relay nodes and one forwarding to the corresponding destination), in a cascaded delivery with k destinations. In the proposed scheme, as relays interact with each other and synchronize in terms of all needed messages, each source or originating node should copy their content to at most L/k relay nodes and let these relay nodes interact with each other to collect all necessary information for final message computation. This will ensure that, if the delivery to the final destination does not happen early, at most the same number of forwardings will be made. Some relays will have computed final message earlier and delivery will happen if they meet the final destination before other relays collect all the information, yielding a smaller cost.

IV. Simulations

To evaluate the proposed scheme, we simulated a DTN with 100 nodes moving in an area of 300×300 m^2. The mobility of nodes is determined by random walk mobility model. That is, first a random direction is selected between [0, 2π] and a random speed is assigned in range of [4, 13] m/s. Then, the node moves in that direction with the assigned speed for a randomly selected duration of [8, 15] s. When the nodes arrive the region boundaries, they are bounced back with the same angle. The wireless transmission range of the nodes is set to 10 m, and if they are in range of other nodes, they are assumed to exchange messages between each other.

We used a cascaded delivery scenario with $k = 3$ destinations. For sequential and joint delivery, we used $L = 5$ relays (including the originating nodes) for each message. For the proposed INC delivery, we used $\lceil 5/3 \rceil = 2$ relays. This means each originating node makes only 1 additional copy of its message, and we let that relay and the originating node interact with other message relays to synchronize their contents. For the PHE calculations, we used 1024-bit primes for p and q defined in the Paillier cryptosystem. With these settings, homomorphic operations take less than 10 ms on a computer with 2.5 Ghz.

We measured the average packet delivery ratio, delay and the number of message forwardings for each delivered packet. Fig. 6 and Fig. 7 show the comparison of average delivery ratios and delay for the three routing approaches: INC delivery, joint delivery, and sequential delivery. The proposed INC delivery scheme achieves much better delivery ratio and less delay compared to sequential delivery. Moreover, it can achieve as high delivery ratio as joint delivery if the delivery deadline (or time-to-live (TTL) of the messages) is high enough. The joint delivery releases the content of the individual messages to the final destination, and thus is not desired. INC delivery avoids that problem and achieves comparably similar high delivery ratio. The cost of all three algorithms were around 15 forwardings. Thus, INC delivery provides, at short delivery deadlines (TTL $<$ 200 s), a practical tradeoff by attaining a privacy-preserving delivery ratio close

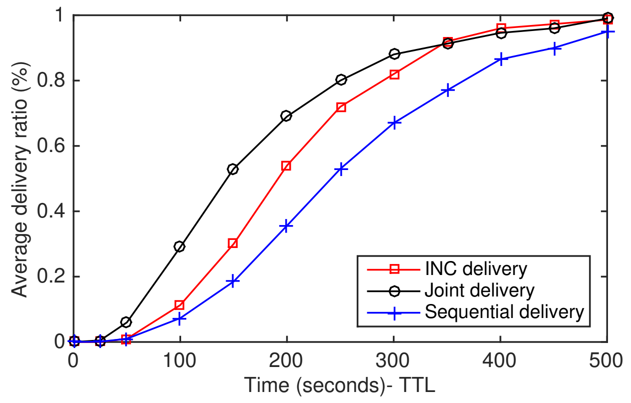

Fig. 6: Average message delivery ratio of In-Network Computed (INC), joint and sequential delivery in a cascaded delivery scenario with $k = 3$ destinations.

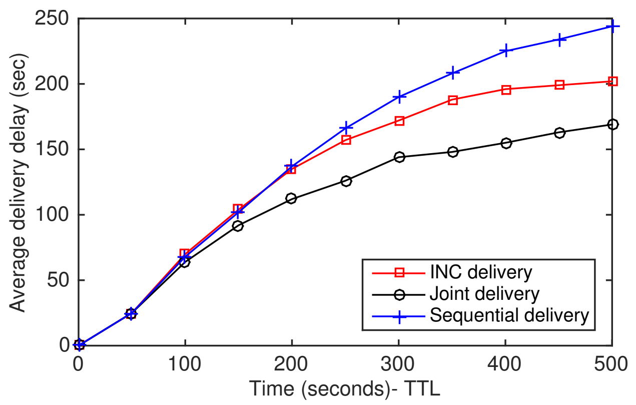

Fig. 7: Average delivery delay comparison of all algorithms.

to the ideal method of joint delivery while causing a minimal additional delivery delay due to the in network computing. At large delivery deadlines (TTL > 350 s), which is more prominent in DTNs), INC attains better delivery ratio while keeping the delivery delay close to the joint delivery.

V. Conclusion

We studied the cascaded delivery problem in a chain of source destination pairs. We proposed an efficient and secure cascaded delivery process using the in-network interaction of relay nodes and computing the content of the final message at relay nodes exploiting the benefits of homomorphic encryption. Our initial results showed the potential for achieving higher delivery ratio while preserving privacy of earlier messages in the cascade chain, i.e., without releasing the messages earlier in the chain to the final destination but only the final message.

In future work, we will look at the results with different settings, such as different number of destinations and heterogeneous node relations. We will also apply the proposed idea in social-based routing algorithms [7], [9] for better cascaded delivery performance. We assumed that the ids of all destinations are known in advance. However, in practice the decision of next destination could be performed at previous destination based on the content of the received message. For example, depending on the emergency situation report from the victim, the request may be directed to the agents providing different service in the area. In future work, we will look at this problem as well.

Acknowledgment

This work was supported in part by NSF awards 1647189 and 1647217.

References

[1] K. Fall, “A delay-tolerant network architecture for challenged internets,” in *Proc. of the 2003 conf. on Applications, technologies, architectures, and protocols for computer commun.* ACM, 2003, pp. 27–34.

[2] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Efficient routing in intermittently connected mobile networks: The multiple-copy case,” *IEEE/ACM Transactions on Networking*, vol. 16, no. 1, pp. 77–90, 2008.

[3] E. Bulut, Z. Wang, and B. K. Szymanski, “Cost-effective multiperiod spraying for routing in delay-tolerant networks,” *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1530–1543, 2010.

[4] E. Bulut and B. K. Szymanski, “Secure multi-copy routing in compromised delay tolerant networks,” *Wireless personal communications*, vol. 73, no. 1, pp. 149–168, 2013.

[5] C. Liu and J. Wu, “On multicopy opportunistic forwarding protocols in nondeterministic delay tolerant networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1121–1128, 2012.

[6] E. M. Daly and M. Haahr, “Social network analysis for routing in disconnected delay-tolerant manets,” in *Proceedings of ACM MOBIHOC*. ACM, 2007, pp. 32–40.

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

[8] E. Bulut, Z. Wang, and B. K. Szymanski, “Time dependent message spraying for routing in intermittently connected networks,” in *Proceedings of IEEE GLOBECOM*, 2008, pp. 1–6.

[9] E. Bulut and B. K. Szymanski, “Exploiting friendship relations for efficient routing in mobile social networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 12, pp. 2254–2265, 2012.

[10] Y. Cao and Z. Sun, “Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges,” *IEEE Communications surveys & tutorials*, vol. 15, no. 2, pp. 654–677, 2013.

[11] W. Gao, Q. Li, B. Zhao, and G. Cao, “Social-aware multicast in disruption-tolerant networks,” *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 5, pp. 1553–1566, 2012.

[12] S. Ding, X. He, J. Wang, and J. Liu, “Pre-decoding recovery mechanism for network coding opportunistic routing in delay tolerant networks,” *IEEE Access*, vol. 6, pp. 14 130–14 140, 2018.

[13] E. Bulut, Z. Wang, and B. K. Szymanski, “Cost efficient erasure coding based routing in delay tolerant networks,” in *2010 IEEE International Conference on Communications*. IEEE, 2010, pp. 1–5.

[14] S.-K. Yoon and Z. J. Haas, “Application of linear network coding in delay tolerant networks,” in *Proc. of IEEE International Conference on Ubiquitous and Future Networks (ICUFN)*, 2010, pp. 338–343.

[15] F. De Pellegrini, R. El-Azouzi, and F. Albini, “Interplay of contact times, fragmentation and coding in dtns,” in *P. of IEEE Int. Symp. on Mod. and Opt. in Mobile, Ad Hoc & Wireless Nets. (WiOpt)*, 2013, pp. 580–587.

[16] F. Jamil, A. Javaid, T. Umer, and M. H. Rehmani, “A comprehensive survey of network coding in vehicular ad-hoc networks,” *Wireless Networks*, vol. 23, no. 8, pp. 2395–2414, 2017.

[17] S. Ahmed and S. S. Kanhere, “Hubcode: hub-based forwarding using network coding in delay tolerant networks,” *Wireless Communications and Mobile Computing*, vol. 13, no. 9, pp. 828–846, 2013.

[18] Z. Zhang, H. Zhang, H. Dai, X. Chen, and D. O. Wu, “Fountain-coded file spreading over mobile networks,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 10, pp. 6766–6778, 2017.

[19] M. Naehrig, K. Lauter, and V. Vaikuntanathan, “Can homomorphic encryption be practical?” in *Proceedings of ACM Workshop on Cloud Computing Security Workshop*, 2011, pp. 113–124.

[20] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 223–238.