Homomorphic Cryptography-based Privacy-Preserving Network Communications

Antoine Guellier, Christophe Bidan, Nicolas Prigent

Homomorphic Cryptography-based Privacy-Preserving Network Communications*

Antoine Guellier, Christophe Bidan, and Nicolas Prigent

CIDre Team, SUPELEC/Inria/CNRS/Université de Rennes 1/IRISA
Supélec Rennes, Cesson-Sévigné, FRANCE
{first.lastname}@supelec.fr

Abstract. The development of worldwide communications, along with the dramatic increase of computational and storage capacities promise numerous benefits for societies in the years to come. For these systems to develop, security and privacy are necessary. This work presents a novel protocol for privacy preserving network communications, using homomorphic cryptography. The malleability properties of homomorphic encryption allows routing without ever disclosing the sender or receiver of a message, while resisting against basic end-to-end attacks. We first present our protocol in an abstract network model, and instantiate it for ad-hoc networks as a use-case example.

Keywords: Privacy, Anonymous, Network, Communications, Routing, Protocol, Homomorphic Cryptography

1 Introduction

Ubiquitous computing and worldwide communications are now part of the daily life of many individuals. We expect from these systems availability and security in order to provide reliable, confidential, and authenticated communications. Since the Snowden revelations [12], we also expect these systems to provide strong privacy. Nowadays, privacy preserving solutions are being adopted by a wider public. The Tor network [8] is the most striking example, with more than 2 million users all over the world.

In general, in any information system, anyone can rightfully require her actions to stay private with respect to a set of chosen individuals. However, in a censored or totalitarian regime, a nonconformist reporter *needs* privacy to exercise her freedom of speech and communicate with the outside. Indeed, the simple fact of connecting to a blacklisted website or participating in a network exposes her to serious consequences [4]. We focus on private communications within contexts where privacy is a necessity.

* An extended abstract of this paper is published in the proceedings of the 5th International Conference on Applications and Techniques in Information Security - ATIS 2014. This is the full version. The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-662-45670-5_15

Anonymity and privacy for network communications begin with Chaum's MIX-nets [2]. They inspired a large portion of works in this field [7], although some may differ by modifying the number of MIX nodes or points in their functioning. Solutions introducing *high latency* between the sending of a packet and its reception are in general more resistant to end-to-end attacks and traffic analysis than their *low latency* counterparts. On the other hand, they are not suited for time-critical or interactive applications such as web surfing or SSH. For instance, Tor [8] and more generally onion routing is a low-latency protocol, while Mixminion [6] or MorphMix [20] are protocols aiming at resisting against end-to-end attacks and thus induce high latency.

Our main contribution is to propose a routing protocol that allows strongly private bidirectional interactive communication using homomorphic cryptography. To the best of our knowledge, it is the first routing protocol relying on this technology. By distributing the trust and using a *proactive topology dissemination*, our protocol resists both internal and external passive adversaries. In addition, our protocol not only hides the relation between senders and receivers, but also decorrelates every message from its sender and receivers everywhere in the network. We further provide an analysis of our protocol in terms of privacy, and study its integration in a real-world example.

Using homomorphic cryptography is motivated by the limitation of traditional cryptography. Indeed, homomorphic encryption (HE) schemes allows to publicly compute the encrypted value $\mathsf{HEnc}(f(m_1, m_2, \ldots, m_k))$ for some function f using the individual encryptions $\mathsf{HEnc}(m_i)$ for $i \in [0..k]$ [21]. Because they are sufficient for our purposes, we solely consider *partial* HE schemes as Paillier or Elgamal [11].

The rest of the paper is organized as follows. In Sect. 2 we present the required tools, our models, and privacy goals. The protocol itself is presented in Sect. 3, while Sect. 4 discusses its properties and further improvements. Before concluding, Sect. 5 investigates its integration in ad-hoc networks [3].

2 Preliminaries

2.1 Cryptographic tools

Homomorphic encryption. This work makes extensive use the Elgamal cryptosystem [10], for its good efficiency compared to other partial schemes and its scalar exponentiation capacity. To make this cryptosystem secure, it must work within a group $\mathbb{G}$ where the Decisional Diffie-Hellman (DDH) assumption holds, *i.e.* in a group of prime order q. Typically, $\mathbb{G} \subset \mathbb{Z}_p^*$ is a subgroup of the multiplicative group of prime order $p = k.q+1$ for some integer k. Keys are of the form $(pk, sk) = (h = g^x, x) \in \mathbb{G} \times \mathbb{Z}_q$ where $\mathbb{G} = \langle g \rangle$. The encryption of a message $m \in \mathbb{G}$ is $c = \mathsf{HEnc}(pk, m) = (g^r, m.h^r) \in \mathbb{G}^2$ for some random value $r \in \mathbb{Z}_q$. Elgamal supports multiplication of ciphertexts $\mathsf{HEnc}(pk, m_1) \times \mathsf{HEnc}(pk, m_2) = (g^{r_1+r_2}, m_1.m_2.h^{r_1+r_2}) = \mathsf{HEnc}(pk, m_1.m_2 \mod p)$, scalar multiplication $\mathsf{ScMult}(m_1, \mathsf{HEnc}(pk, m_2)) = (g^r, m_1.m_2.h^r) = \mathsf{HEnc}(pk, m_1.m_2 \mod p)$, and scalar exponentiation $\mathsf{ScExp}(\mathsf{HEnc}(pk, m_1), m_2) = (g^{r.m_2}, m_1^{m_2}.h^{r.m_2}) = \mathsf{HEnc}(pk, m_1^{m_2}$

mod p). Elgamal also supports *re-randomisation*: a ciphertext $c = (c[0], c[1])$ for some message can be turned into a different looking, randomly distributed ciphertext for the same message: $\mathsf{ReRand}(pk, c) = (c[0].g^{r'}, c[1].h^{r'})$.

Threshold homomorphic encryption. This work also uses a *threshold homomorphic encryption* (THE) scheme [5]. This type of homomorphic scheme can have its secret keys split into parts and distributed among several entities. As a result, for some threshold t, t entities or more can collaboratively decrypt ciphertexts, but less than t entities do not learn anything on the underlying plaintexts. In practice, key generation is performed by an authority who publicizes the public key and distributes shares of the secret key. Encryption is performed as usual, but decryption requires at least t share holders: each one locally decrypts the ciphertext using its share, and the outputs are then combined to recover the plaintext. Although any THE scheme may be suitable, this work uses the threshold variant of Paillier from Damgård *et al.* [5]. We denote encryption, decryption using a share, and combining primitives of a THE scheme by $\mathsf{THEnc}(pk, m)$, $\mathsf{THDec}(share, c)$, and $\mathsf{THComb}(\hat{c}_1, \ldots, \hat{c}_t)$.

Note that in the rest of the paper, standard public-key encryption (PKE) is denoted by Enc, and symmetric-key encryption using key K is denoted $\{\cdot\}_K$.

2.2 Models and Goals

In the rest of the paper, we denote the node initiating a connection *source*, and its intended receiver *destination*. We should however mention that once the connection is established, information may flow in both directions. Nodes between a source and a destination *relay* the messages on a multi-hop path. We call a *message* any data transiting through the network, whether it is application-level data or routing information.

Distributed trust. We rely on a *distributed trust* model, where any node may initiate, relay or be the receiver of a connection. By distributing the routing and removing all hierarchy among the nodes, we limit the consequences of node corruption. As there is no central network entity nor dedicated relays, corrupting a node brings very little power on the network as a whole. Actually, in our model, a node is unable to make use of the routing information on its own to run the protocol: it must collaborate with other network member. This suggests a corrupted node alone can not break the privacy of our protocol.

Privacy Goals. We refer to Pfitzmann and Köhntopp [19] to define the concept of *unlinkability* in our setting: two *items of interest* are unlinkable if the adversary is unable to distinguish whether they are related or not. Using this terminology, we aim at ensuring the following properties.

Source/message unlinkability (SMU) Given a message m and a node S, it is impossible to distinguish whether S is the source of m or not;

Destination/message unlinkability (DMU) Given a message m and a node D, it is impossible to distinguish whether D is the destination of m or not;

Source/destination unlinkability (SDU) Given two nodes, it is impossible to distinguish whether they communicated or not;

Message/message unlinkability (MMU) Given two messages, it is impossible to distinguish whether they belong to the same source-destination pair communication or not;

Adversary Model. We consider two adversary models: a *passive*, *non-collusive* adversary *internal* to the network (also called *honest-but-curious*) (AdvI), and a global eavesdropper (AdvII). The first type of adversary participates in the network and is compliant with the protocol, but tries to get as much knowledge as possible on the network and other nodes. The second is still passive but more powerful, although it does not participate in the network. We do not give the adversary the ability to tamper with the network by dropping, replaying, or performing DoS or Sybil attacks for instance. Note that when considering information theoretic adversaries, notably against cryptographic primitives, we use the probabilistic polynomial-time (PPT) model. In any case, the goal of the adversary is to break the properties stated above.

Non-goals. We do *not* intend to provide an implementation-ready, nor an efficient protocol. Our first goal is to design a routing protocol for interactive communications achieving as much privacy as possible, and to extract interesting properties from it. Improvements in terms of efficiency and routing optimizations belong to future work. At last, protecting messages at the application level, although necessary, is outside the scope of this work. In particular, to resist against fingerprinting techniques on the Internet[17], content and identity should be protected end-to-end using SSL, application-level anonymization proxy or similar tools.

3 The protocol

Bearing in mind the desired privacy properties, we aim at designing a protocol that constructs and uses routes so that any node can contact any others at any time. This section describes our solution in two steps: it first shows how to disseminate topology without using nodes' identities, and then how the routing information is used to allow message sending.

3.1 Infrastructure

Initially, we consider an abstract network model where nodes are randomly placed and connected to their neighbors through a bare physical medium. Prior to describing our protocol, we introduce several values associated to a node X.

- ID_{X}: X's network identity. It is a random value chosen by X prior to network setup, typically extracted from a large public set so that collisions between two nodes' identifiers occur with negligible probability. Generally speaking, nodes are willing to keep their identity concealed to other network members.

- $(pk_{\mathrm{X}}^{\mathrm{perm}}, sk_{\mathrm{X}}^{\mathrm{perm}})$: the permanent key pair of X, possibly certified or publicly linked to X's identity.
- $(pk_{\mathrm{X}}, sk_{\mathrm{X}})$: one of X's *temporary* key pairs, decorrelated from X's identity. A node may own many temporary keys and dynamically generate them.
- src_{X}: a value generated by X and used in its routing activity as a source (or relay) node. This value is unconditionally kept secret by X.
- dst_{X}: a value generated by X and used in its routing activity as a destination node. This value is unconditionally kept secret by X.

We assume that prior to network setup, nodes willing to communicate exchanged their respective network identifiers, long term public keys, and possibly some auxiliary information, in the same manner as PGP keys are exchanged. In a very constrained context such as a censorship, a very high latency but highly private and secure protocol allowing the transport of small data, such as MIAB [14], can be used.

3.2 Routing while concealing identities

As we aim at concealing every node's identity, the main challenge we have to face is to transport a message to its intended recipient without knowing its actual identity. More exactly, we need to find a solution to *route without network members knowing other nodes' network-wide identities*. For this, we introduce the notion of *local identifiers*: instead of using a destination D's identity ID_{D} to designate it, a node X uses a value derived from values belonging to D and itself. This value is collaboratively computed by X and D upon creation of the route. It should allow X to route messages towards D, but must prevent X from making the link with ID_{D}. We denote this value $LocalID_{\mathrm{D}}^{\mathrm{X}}$ and define it as:

$$LocalID_{\mathrm{D}}^{\mathrm{X}} = ID_{\mathrm{D}}.g^{src_{\mathrm{X}}.dst_{\mathrm{D}}}, \text{ where } \langle g \rangle = \mathbb{G}$$

Because most of our operations are performed *within* the Elgamal group $\mathbb{G} \subset \mathbb{Z}_p^*$, the local identifiers and more generally all values in our protocol are computed modulo p (unless specified otherwise).

The local identifiers resolve the question of destination addressing. However, we also need a way for node to forward messages towards some local identifier on a multi-hop path. Indeed, message must travel from node to node in order to reach its destination. This implies that nodes must be able to address specific neighbors, still without knowing their identities. To enable private neighborhood discovery, we borrow a technique from Zhang *et al.* [22]. It consists, for every pair of neighboring node in the network, in engaging in a Diffie-Hellman (DH) handshake in order to obtain a common secret, and to derive symmetric keys and *link identifiers* from this secret using cryptographic key derivation functions. A link identifier between two nodes X and Y, denoted $L_{\mathrm{X,Y}}$ or $L_{\mathrm{Y,X}}$ indifferently, acts as a MAC address in a traditional LAN. The key $K_{\mathrm{X,Y}}$ associated to $L_{\mathrm{X,Y}}$ serves to encrypt the message on the link. The particularity of this construction is that link identifiers and keys are only used one time. New ones are generated

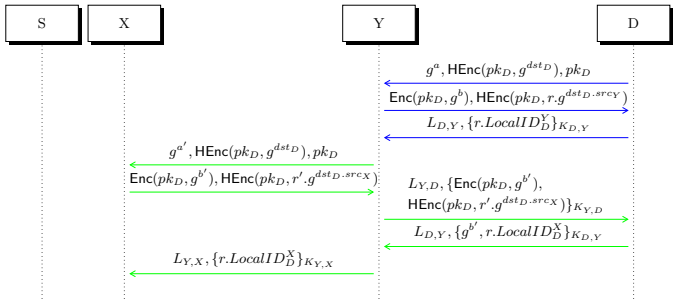

Fig. 1. Topology dissemination in a toy network.

for each message (without the need for a new handshake), in such a way that it is impossible to relate two links (or keys) stemming from the same handshake.

We now describe how to disseminate the topology without revealing any node identity. The general idea works as follows: when a node knows a route, it *proposes* this route to its neighbors. Said otherwise, it offers its neighbors to forward messages towards some destination for them. We make use of the elements defined above, and store routing information into two different tables: a *Destination Route Table* (DRT) and a *Forwarding Route Table* (FRT). To explain how these tables are filled, we use the example from Fig. 1 depicting a toy network of 4 nodes in a row.

In a first phase (in blue), D broadcasts to all its neighbors a *route proposal*: it proposes itself as destination. The goal of this route proposition is to make neighboring nodes aware of D and compute their local identifiers towards D. For this, the neighboring node Y generates a random number r, and computes $\mathsf{ScMult}\left(r,\ \mathsf{ScExp}(\mathsf{HEnc}(pk_D, g^{dst_D}), src_Y)\right)$ using the homomorphic properties of Elgamal. It sends a re-randomization of the result so as to ensure the ciphertext looks random, and a second half of a DH handshake (encrypted, to avoid *man-in-the-middle* attacks). D decrypts the ciphertext, computes $ID_D.r.g^{dst_D.src_Y} = r.LocalID_D^Y$ and sends the result back using the newly generated link identifier and key. Y recovers $r.LocalID_D^Y$ and simply multiplies by $(r^{-1} \mod p)$. After these exchanges, D inserts a new entry in its FRT: $\langle L_{D,Y}, K_{D,Y}, null\rangle$. The *null* value indicates the end of a route, and allows D to know that it is the intended receiver of messages incoming from the link $L_{\mathrm{D,Y}}$. As for Y, it inserts an entry in its DRT: $\langle LocalID_D^Y, L_{Y,D}, K_{Y,D}, pk_D, \mathsf{ReRand}(pk_D, \mathsf{HEnc}(pk_D, g^{dst_D}))\rangle$, where the last value is simply a re-randomized copy from D's first message.

In the second phase (in green), Y proxies the route proposal from D to its own neighbors. D silently discards the proposal upon noticing it is the destination. The protocol is then very similar, except that the handshake is handled somewhat differently, and that Y must act as proxy between X and D because D is the only node able to decrypt X's answer. After the exchanges, Y inserts in its FRT $\langle L_{X,Y}, K_{X,Y}, LocalID_D^Y\rangle$, and X insert in its DRT $\langle LocalID_D^X, L_{X,Y}, K_{X,Y}, pk_D, \mathsf{ReRand}(pk_D, \mathsf{HEnc}(pk_D, g^{dst_D}))\rangle$.

Local id	*Link id*	*Key*	*Dest. pk*	*Route prop.*
$LocalID_D^Y$	$L_{Y,D}^{(1)}$	$K_{Y,D}^{(1)}$	pk_D	$\mathsf{HEnc}(pk_D, g^{dst_D})$
$LocalID_X^Y$	$L_{Y,X}^{(1)}$	$K_{Y,X}^{(1)}$	pk_X	$\mathsf{HEnc}(pk_X, g^{dst_X})$
$LocalID_S^Y$	$L_{Y,X}^{(2)}$	$K_{Y,X}^{(2)}$	pk_S	$\mathsf{HEnc}(pk_S, g^{dst_S})$

Destination Table

Link id	*Key*	*Local id*
$L_{Y,D}^{(2)}$	$K_{Y,D}^{(2)}$	*null*
$L_{Y,X}^{(3)}$	$K_{Y,X}^{(3)}$	*null*
$L_{Y,D}^{(3)}$	$K_{Y,D}^{(3)}$	$LocalID_X^Y$
$L_{Y,D}^{(4)}$	$K_{Y,D}^{(4)}$	$LocalID_S^Y$
$L_{Y,X}^{(4)}$	$K_{Y,X}^{(4)}$	$LocalID_D^Y$

Forwarding Table

Link ids with different parenthesized exponents stem from two different DH handshakes.

Fig. 2. Routing tables of Y after topology dissemination

This procedure is repeated, X proposing the route to S and X and Y proxying S's answer. In the general case, the procedure is repeated until the whole network is aware of D's presence. Furthermore, all nodes self-propose in the same manner as D to advertise their presence. As a result, each node's presence in the network will eventually be known by all, while its identity remains undisclosed. In our example, Y's tables are eventually filled as described in Fig. 2.

Using regular timeouts for table entries, and by repeating the proposals periodically, tables can be updated in case a node leaves or moves in the network. Although the complexity of this topology dissemination method is considerable (at least quadratic in the number of nodes in the network), it is only necessary to run it once at network setup.

3.3 Sending a message

At this point, each node has at least one route towards every other node. Note that completely anonymous communications are possible: the two tables allow any two nodes in the network to connect to each other using local identifiers. However, nodes can not know who they are connecting with, and the tables do not directly allow a source to reach a specific destination of its choice.

In order to allow connection of S to D without breaching privacy, S and D not being 1 hop neighbors, our solution makes S ask for help to another node. Also, we use a THE scheme with threshold $t = 2$ and make the assumption that there exist a pair of keys (pk_*, sk_*) for this scheme such that pk_* is publicly known, but sk_* is unknown to all network members. However, every node X owns a share of the decryption key, sh_X. The key pair and shares can be generated by a third party for instance. Finally, we suppose that, in addition to its identity and public key, D gave S the auxiliary value $\mathsf{THEnc}(pk_*, dst_\mathrm{D})$.

Figure 3 describes the message sequence for S to be able to send a message msg to D. The first phase (in blue) is called *route initialization*: this is the part where S and X cooperate using THE to find a route from X to D[1]. Before running this protocol, S generates a random number r and computes $C = \mathsf{ReRand}(pk_*, \mathsf{ScMult}(r, \mathsf{THEnc}(pk_*, dst_\mathrm{D})))$. S and X then use their respective decryption shares sh_S and sh_X to cooperatively decrypt C. X obtains a

[1] In the general case, S can choose any node in the network to play the role of X.

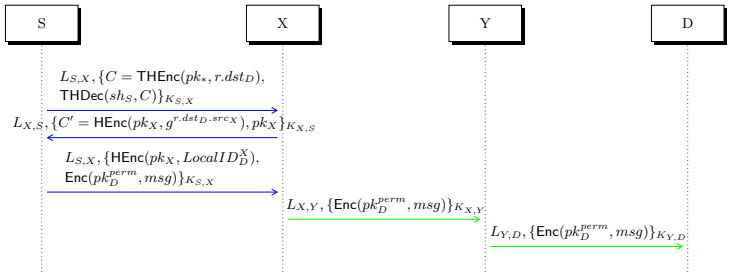

Fig. 3. S sends a message to D.

blinded value of dst_D, $v = r.dst_\mathrm{D}$, by using the THComb primitive. X then computes in the clear $(g^{src_\mathrm{X}})^v$, encrypts it under one of its keys using Elgamal, and sends back the result C'. If r and $p-1$ are co-prime, applying $\mathsf{ReRand}(pk_\mathrm{X}, \mathsf{ScMult}(ID_\mathrm{D}, \mathsf{ScExp}(C', r^{-1} \mod (p-1))))$ on the ciphertext C' received by S yields a random-looking encryption of $LocalID_\mathrm{D}^\mathrm{X}$, which X is able to decrypt. X looks up in its DRT for a route towards this local identifier, and forwards the payload encrypted under D's permanent key. The second phase (in green) simply consists in table look ups. Upon receiving the message, Y looks up in its FRT to find the entry corresponding to $L_{\mathrm{X,Y}}$. Y recovers the relevant local identifier from this entry, looks up in its DRT for a route toward it, and forwards accordingly. If at any moment, a table look up fails, a notification is sent back. D knows it is the destination thanks to the *null* value in its FRT entry corresponding to $L_{\mathrm{Y,D}}$.

4 Discussion

This section explains our technical choices, analyses our protocol, and discusses its properties.

4.1 Interaction and THE in route initialization

In our model, the interaction in the route initialization is necessary in order to ensure DMU. Indeed, because of the way routes are built, the same local identifiers may be used both to emit or to forward a message to a given destination. Suppose a source S could, on its own, find a route towards a specific destination D, *i.e.* S can find the value $LocalID_\mathrm{D}^\mathrm{S}$ in its DRT and knows it actually designates D. Later on, if S is asked to forward a message towards the same destination D, S will use the value $LocalID_\mathrm{D}^\mathrm{S}$ (as explained in Sect. 3.3). Thus, S will infer the message is for D and break DMU. Therefore, for DMU to hold, a source S *must not* be able to initiate a communication alone.

The most straightforward solution is for S to hand all its messages to some other node X, and for X to forward messages towards the actual destination D. This means that X must find the relevant $LocalID$ to send the messages to, but for DMU to hold, it must not uncover the actual intended destination. For the

same reason, S must not make the link between ID_{D} and $LocalID_{\mathrm{D}}^{\mathrm{S}}$. In other words, through an oblivious interaction with S, X must learn $LocalID_{\mathrm{D}}^{\mathrm{X}}$.

We argue that to achieve secure computation of $LocalID_{\mathrm{D}}^{\mathrm{X}}$, a strong assumption need to be done: every node must possess a *universal auxiliary information* on the network. That is to say, a value that does not relate to a particular node, but on the contrary relates to *all* nodes. We actually show in appendix of this paper, that without such auxiliary information, an efficient and secure route initialization is impossible in our model[2]. In a nutshell, if we do not use universal auxiliary information, we show that, because X can not use any knowledge it has on D (else X would know the intended destination beforehand, which is absurd), X does not bring any input to the computation of $LocalID_{\mathrm{D}}^{\mathrm{X}}$ except src_{X}. This means that S must know the rest of the inputs necessary to compute it, *i.e.* S knows ID_{D} and dst_{D}. Therefore, S can compute $LocalID_{\mathrm{D}}^{\mathrm{S}}$ by itself, and break DMU as explained above. The proposed solution is to give the power to S and X to compute *together* something they could not compute by themselves. We achieve this *via* a piece of information that each node keeps secret, different for each node and useless by itself, but when several pieces are combined, they produce a necessary input to all $LocalID$ values. In practice, we instantiate these auxiliary information by shares of a secret key of a THE scheme.

4.2 Privacy analysis

Our protocol, as described in the previous section, is still vulnerable to many end-to-end attacks and traffic analysis. To prevent them, we fix a constant message size[3] and we borrow techniques from MIX networks such as the introduction of random delays and message batching [7]. These techniques unfortunately incur a loss of performances under the form of additional latency. We reject the insertion of dummy messages, for their prohibitive cost compared to the additional privacy they provide [18].

Preliminary properties Intuitively, our design ensures the following properties:

(1) By construction, *routes are partially disjoint*: for a given source-destination pair there may exist several routes, and for two different source-destination pairs, routes may overlap. Thus, in Fig. 4, Z can not know whether messages come from S_1 or S_2, as the link identifier is the same for both routes.
(2) If the Decisional DH assumption holds and secure cryptographic key derivation functions exist, link identifiers are *secure*: for a given pair of neighboring nodes (X, Y), link messages are unlinkable for AdvI and AdvII. That is to say, the pairs $(L_{\mathrm{X,Y}}^{(1)}, K_{\mathrm{X,Y}}^{(1)})$ and $(L_{\mathrm{X,Y}}^{(2)}, K_{\mathrm{X,Y}}^{(2)})$ stemming from two different DH handshakes are unlinkable.

[2] There is a way to achieve route initialization by performing an *exhaustive search* in both S and X's DRTs, but this yields a quadratic complexity in the number of nodes.
[3] Large messages are split into several ones, and padding is used if necessary.

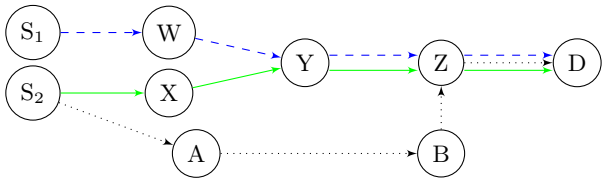

Fig. 4. An example of partially disjoint routes

(3) The local identifiers are *secure*: for all X and D, $LocalID_{\mathrm{D}}^{\mathrm{X}}$ is known only to X; if the Computational DH assumption holds, given $LocalID_{\mathrm{D}}^{\mathrm{X}}$ it is not possible to recover the inputs that formed it; and if the DDH assumption holds, for any X, D_1, D_2, $LocalID_{D_1}^{X}$ and $LocalID_{D_2}^{X}$, X is unable to know which local identifier points to which destination.
(4) Route propositions are *indistinguishable* for AdvI. Because, locally, the messages exchanged are the same in both cases, whether a route proposition emanated from the destination itself or a proxy is indistinguishable for AdvI.
(5) If route propositions are *indistinguishable* for AdvI, if the encryption schemes used are semantically secure, if the link identifiers are secure, and if we use MIXing techniques in the route proposal, then up to a certain point, AdvII can not distinguish whether a proposition emanated from the destination or a proxy[4].
(6) If local identifiers are secure, route initialization is *oblivious* for AdvI. Indeed, S never makes the link between D and its value $LocalID_{\mathrm{D}}^{\mathrm{S}}$, and V never learns that the destination is D. In addition, the blinding factor r hides dst_{D} and if the discrete logarithm problem is hard, src_{X} is concealed.
(7) If we use MIXing techniques, and if link identifiers are secure, route initialization is indistinguishable from standard communication for AdvII.

Proof sketches of these properties are available in appendix. We merely note here that property 3 seem to hold for the same reason that Elgamal is semantically secure. Indeed, $LocalID_{\mathrm{D}}^{\mathrm{X}}$ can be seen as an Elgamal encryption of ID_{D} under key dst_{D} using randomness src_{X}. However, the proof can not be the same, as in our case, the random coins src_{X} are known to X.

Privacy properties Assuming the properties stated above hold, our protocol fulfills our privacy goals. We provide proof sketches in appendix.

- **SMU** holds against
  - **AdvI**, because messages are exempt from any direct or indirect information on the source or its location in the network.
  - **AdvII**, if we introduce MIXing techniques in the forwarding process and properties 2 and 7 hold.
- **DMU** holds against

[4] Up to a certain point, because when the proxy is far from the destination, a basic timing analysis is sufficient to distinguish the two cases.

- **AdvI**, if properties 3, 4 and 6 hold, and if the encryption schemes are receiver-anonymous [15].
- **AdvII**, if properties 2 and 5 hold, and if the destination, upon receiving a message, simulates a relay.

– **SDU** holds against **AdvI** and **AdvII** if SMU or DMU hold.
– **MMU** holds against
- **AdvI**, if property 1 holds. Except in the special case of node X in Fig. 3.
- **AdvII**, if property 2 holds.

4.3 Shortcomings and future improvements

The above analysis is mainly intuitive. Formal proofs are desirable in order to confirm that the privacy goals are met. For this, the framework of Hevia and Micciancio [13] provides a formal definition of source, destination and/or message unlinkability against adversary models that are very close to ours.

Because we mainly focused on privacy, our solution is neither efficient nor optimized. In particular, it does not give any guarantee on the connectivity of the network, it does not manage route lifetimes, does not handle routing loops and does not limit the size of the routes. For a practical routing algorithm, mechanisms handling those points are necessary. The difficulty is to design such mechanisms while preserving privacy.

Our protocol is vulnerable to active attacks. To port security against *active internal non-collusive* adversaries, it has to be modified so that homomorphic evaluations performed by one party can be *verified* by another. For this, one may use *homomorphic message authentication codes*, which certify that some ciphertext stem from the correct homomorphic computation [1].

Also, our protocol is vulnerable to a collusion of merely 2 nodes. For instance, if S and X collude in the route initialization protocol, they uncover the secret value dst_{D} of D and S eventually break DMU. To resist against a collusion of t nodes, we can ask the relay nodes between S and X (assuming X is not neighbor to S) to participate in the threshold decryption protocol.

It would also be interesting to remove the strong assumption of universal auxiliary information. Indeed, although it is acceptable for particular contexts, assuming that each node owns a share of a common secret is unrealistic in general. As we have seen, in our solution, this hypothesis is necessary to obtain an efficient route initialization. A solution is to distribute the secret and secret shares generation, but in this case, the dynamic insertion of nodes is very complex. Other choices, models or assumptions might allow circumventing the impossibility result while enabling efficient route initialization. For instance, decorrelating the activities of sources and relays, so as to avoid issues raised in Sect. 4.1, would remove the need for interaction and universal auxiliary information.

At last future work should focus on testing the protocol against typical network attacks, and to assess the practical usability of our design measuring efficiency, available bandwidth and scalability through simulations.

5 Adaptation to a real-wold example: ad-hoc networks

We defined our protocol in a general, abstract network. As in our model, a simple physical medium is necessary, our protocol can work on top of any bare network graph. Indeed, no infrastructure nor any specific routers are required for our protocol to be deployed. Now, this section investigates how our protocol can be instantiated to a real world example. Because of the similarities with our model, we envision an adaptation to ad-hoc networks [3].

Indeed, these networks are infrastructure-less: there is no central node(s), and the routing is completely distributed. Ad-hoc routing protocols are either *reactive* or *proactive*. In the formers, routes are created only when necessary in an on-demand fashion while the others behave more as IP routing, where topology is flooded in the network. Although anonymous reactive routing designs are overwhelming, proactive ones are quite rare [16]. The few we are aware of either make assumptions we reject such as the presence of nodes willing to disclose their identities, or provide only a weak form of privacy (but better efficiency). Our solution stands out by its proactive nature, by the strong privacy it provides, and by the use of homomorphic cryptography, which has never been used before in ad-hoc networks.

Ad-hoc networks allow a smooth integration of our protocol. Indeed, we can directly work above the MAC layer, assuming that nodes are able to function in *promiscuous* mode and to set their address to the broadcast address. Thus, nodes always use MAC broadcasting to communicate, and link identifiers to address specific neighbors, as described originally by Zhang *et al.* [22]. Thanks to the broadcast nature of ad-hoc networks, privacy is even re-enforced. Indeed, when a node emits a message using link identifiers, AdvII can not even know which neighbor was addressed. This is not the case with wired networks, where links are point-to-point.

If we envision an adaptation to ad-hoc networks, the main shortcoming of our protocol is its inefficiency. The cost of cryptographic primitives is prohibitive for ad-hoc network nodes, which are typically small devices with limited battery power. At last, for an adaptation to *mobile* ad-hoc networks (MANETs), we have to take in account the nodes' mobility, which might cause route exhaustion and imprecise knowledge of the neighborhood. To deal with frequent topology changes, a trivial yet inefficient solution is to increase the frequency of route proposals. Another possibility is to rely on the fact that, as several routes are created from each source-destination pair, there will always be a valid route.

6 Conclusion

This preliminary work lays the foundations for strongly private network communications using homomorphic encryption. We presented a privacy preserving routing protocol suitable for distributed environments, exhibited its main features, and provided a partial proof of its privacy properties. As practical use-case, we showed that an adaptation to ad-hoc networks is at hand.

Many points need to be addressed, and several others can be improved. In particular, future work will include advanced experimentations to attest the gain in privacy provided by our protocol and its efficiency. Even though we do not expect our protocol to perform as good as the state-of-the-art, we assume our users are willing to trade some efficiency against strong privacy (a feature often lacking in presently deployed solutions[5]). If possible, privacy properties should be proved formally, but how to do so is still an open question. Even technologies already widely deployed, which benefit from years of experience, regularly suffer attacks. Of course, we do not intend to replace these solutions, but merely to complete them. We see our work as a proof of concept for new strongly private interactive communications, offering free communications in any circumstances.

Acknowledgment

This work is partially funded by the *Collège International Doctoral de l'Université Européenne de Bretagne* (International Doctoral College of the European University of Brittany, France) and the Regional Council of Brittany, France. The authors would like to thank Simon Boche and Cristina Onete for their insightful suggestions and discussions on various components of the protocol and cryptographic-related topics, as well as for the precious advices brought during the writing of this paper.

References

1. Catalano, D., Fiore, D.: Practical homomorphic macs for arithmetic circuits. In: Johansson, T., Nguyen, P. (eds.) Advances in Cryptology – EUROCRYPT 2013, LNCS, vol. 7881, pp. 336–352. Springer Berlin Heidelberg (2013)
2. Chaum, D.L.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM 24(2), 84–90 (Feb 1981)
3. Chlamtac, I., Conti, M., Liu, J.J.N.: Mobile ad hoc networking: imperatives and challenges. Ad Hoc Networks 1(1), 13 – 64 (July 2003)
4. Cohn, W.A.: Yahoo's china defense. The New Presence 9(3), 30–33 (Sept 2007)
5. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In: Kim, K. (ed.) Public Key Cryptography, LNCS, vol. 1992, pp. 119–136. Springer Berlin Heidelberg (2001)
6. Danezis, G., Dingledine, R., Mathewson, N.: Mixminion: Design of a type iii anonymous remailer protocol. In: Proceedings of the 2003 IEEE Symposium on Security and Privacy. pp. 2–. SP '03, IEEE Computer Society, Washington, DC, USA (2003)
7. Diaz, C., Preneel, B.: Taxonomy of mixes and dummy traffic. In: Deswarte, Y., Cuppens, F., Jajodia, S., Wang, L. (eds.) Information Security Management, Education and Privacy, IFIP International Federation for Information Processing, vol. 148, pp. 217–232. Springer US (2004)
8. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th Conference on USENIX Security Symposium. SSYM'04, vol. 13, pp. 21–21. USENIX Association, Berkeley, CA, USA (August 2004)

[5] A list of works is available at http://www.cs.kau.se/philwint/censorbib/ (thanks to Philipp Winter).

9. Díaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: Dingledine, R., Syverson, P. (eds.) Privacy Enhancing Technologies, Lecture Notes in Computer Science, vol. 2482, pp. 54–68. Springer Berlin Heidelberg (2003)
10. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G., Chaum, D. (eds.) Advances in Cryptology, LNCS, vol. 196, pp. 10–18. Springer Berlin Heidelberg (1985)
11. Fontaine, C., Galand, F.: A survey of homomorphic encryption for nonspecialists. EURASIP J. Inf. Secur. 2007, 15:1–15:15 (Jan 2007)
12. Greenwald, G.: No Place to Hide. Metropolitan Books (May 2014)
13. Hevia, A., Micciancio, D.: An indistinguishability-based characterization of anonymous channels. In: Borisov, N., Goldberg, I. (eds.) Privacy Enhancing Technologies, LNCS, vol. 5134, pp. 24–43. Springer Berlin Heidelberg (2008)
14. Invernizzi, L., Kruegel, C., Vigna, G.: Message in a bottle: Sailing past censorship. In: Proceedings of the 29th Annual Computer Security Applications Conference. pp. 39–48. ACSAC '13, ACM, New York, NY, USA (2013)
15. Kohlweiss, M., Maurer, U., Onete, C., Tackmann, B., Venturi, D.: Anonymity-preserving public-key encryption: A constructive approach. In: De Cristofaro, E., Wright, M. (eds.) Privacy Enhancing Technologies, LNCS, vol. 7981, pp. 19–39. Springer Berlin Heidelberg (2013)
16. Liu, J., Kong, J., Hong, X., Gerla, M.: Performance evaluation of anonymous routing protocols in manets. In: Wireless Communications and Networking Conference, 2006. WCNC 2006. IEEE. vol. 2, pp. 646–651 (April 2006)
17. Nikiforakis, N., Kapravelos, A., Joosen, W., Kruegel, C., Piessens, F., Vigna, G.: Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In: IEEE Symposium on Security and Privacy. pp. 541–555 (2013)
18. Oya, S., Troncoso, C., Pérez-González, F.: Do dummies pay off? limits of dummy traffic protection in anonymous communications. In: De Cristofaro, E., Murdoch, S. (eds.) Privacy Enhancing Technologies, LNCS, vol. 8555, pp. 204–223. Springer International Publishing (2014)
19. Pfitzmann, A., Köhntopp, M.: Anonymity, unobservability, and pseudonymity — a proposal for terminology. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies, LNCS, vol. 2009, pp. 1–9. Springer Berlin Heidelberg (2001)
20. Rennhard, M., Plattner, B.: Introducing morphmix: Peer-to-peer based anonymous internet usage with collusion detection. In: Proceedings of the 2002 ACM Workshop on Privacy in the Electronic Society. pp. 91–102. WPES '02, ACM, New York, NY, USA (2002)
21. Vaikuntanathan, V.: How to compute on encrypted data. In: Galbraith, S., Nandi, M. (eds.) Progress in Cryptology - INDOCRYPT 2012, LNCS, vol. 7668, pp. 1–15. Springer Berlin Heidelberg (2012)
22. Zhang, Y., Liu, W., Lou, W., Fang, Y.: Mask: anonymous on-demand routing in mobile ad hoc networks. Wireless Communications, IEEE Transactions on 5(9), 2376–2385 (September 2006)

A Correctness and performances of arithmetic operations

Our design makes extensive use of Elgamal, and most of the cryptographic operations are performed within the Elgamal group $\mathbb{G}$. For instance, in order to take full advantage of the scheme's homomorphic properties, the local identifiers and the node identities need to belong to the same space as the ciphertexts. This is a non-standard use of

Elgamal: usually, messages belong to $\mathbb{Z}_q$, and a message encoding function from $\mathbb{Z}_q$ to $\mathbb{G}$ is applied to them. In our case, messages are directly taken in $\mathbb{G}$. Furthermore, the arithmetic operations we use often modify both the Elgamal ciphertext and the underlying plaintext. The most representative example is the modular root computation during the route initialization, that takes place *inside* a ciphertext.

To check the correctness of our computations, and more specifically of the ScMult and ScExp functions and the modular root, we developed a simple PHP script[6]. By programming an encoding-free version of Elgamal, we successfully verified that the computation of the local identifiers in the route proposal and initialization protocols yield the right output. The choice of the language was motivated by the facility and rapidity to develop with PHP. The code should not, in any case, be considered secure or be used for any other purpose than checking correctness of the arithmetic operations involved in the protocol.

Although a highly non-optimized coding, we report in Table 1 the timings for a toy instantiation of Elgamal with security parameter $\lambda = 10$, without taking in account any notion of communication delay. The threshold decryption phase is also not represented. Table 2 compares the running times of the C++ (with the GMP library) and the PHP modular exponentiation algorithm on the same test platform. The results show that we can hope augmenting performances by a factor of at least $50,000$ in future implementations.

Table 1. Running times of poof-of-concept PHP implementation for $\lambda = 10$.

KeyGen	Enc	Dec	Ctxt mult.	ScMult	ScExp	ReRand	Rt prop.	Rt init.
47.46ms	0.14ms	0.24ms	0.13ms	0.07ms	0.13ms	0.20ms	1.32ms	1.25ms

Table 2. Comparison of PHP and C++ modular exp. running times for $\lambda = 20$.

C++ (GMP lib)	PHP
0.0008ms	54ms

B Proof of privacy properties

This appendix gives arguments towards proving that our protocol achieves the privacy goals stated in Sect. 4.2, by first showing that the seven properties from the same section hold.

These "proofs" are merely written arguments, sometimes making reference to hard problems in cryptography, such as the discrete logarithm problem. Indeed, although there exists metrics or formal tools for anonymity in network communications, we reject their use here (mainly because our work is preliminary).

In [9], the authors measure anonymity by modeling the knowledge of the adversary with respect to some event (*e.g.* the sending of a message) as a probability distribution

[6] Available at https://github.com/aguellier/elgamal-for-private-comm

on the nodes' identities. This probability distribution is then compared to the uniform probability distribution by computing the difference of their Shanon entropy. As the authors themselves underline, the metric suffers from several shortcomings. First, it gives no indications so as to generating the adversary's knowledge, and although it is straightforward in small examples, this task is uneasy in general. Secondly, the adversary's auxiliary information is not taken into account. For instance, the adversary may know that Alice and Bob participate in the network, and that they are likely to communicate. We add to these facts that setting the threshold of entropy difference above which the privacy is considered broken is unclear. Furthermore, whether the threshold should be chosen depending on the number of nodes, the density of the network, or the number of corrupted nodes is also an open question. At last, in our case, identities are not disclosed at all, so the probability distribution modeling the adversary's knowledge can not be based on them.

Another measure of privacy as been proposed by Hevia and Micciancio under the form of a cryptographic framework [13]. They define privacy properties in a very formal way, and express security through cryptographic games and in terms of *indistinguishability*. Their model seems is close to ours, and their framework seems suited to our setting. However, the authors do not give any clue towards using it. And in the general case, it is unclear how to use their game-based security definition.

For all these reasons, we will settle for rather informal proofs, reasoning at a high level.

B.1 Preliminaries

When possible security properties are formalized *via* game-based security definitions. In those definitions, the adversary is denoted $\mathcal{A}$. The oracle for a function or primitive $f(x)$ is noted$\mathcal{O}_{f(\cdot)}$. An adversary having access to the oracle for function f is denoted $\mathcal{A}^{\mathcal{O}_{f(\cdot)}}$.

B.2 Preliminary properties

This section considers the seven properties from Sect. 4.2 and discusses them one by one.

Property 1 *By construction, routes are partially disjoint: for a given source-destination pair there may exist several routes, and for two different source-destination pairs, routes may overlap. Thus, in Fig. 4, Z can not know whether messages come from S_1 or S_2, as the link identifier is the same for both routes.*

This property simply follows from observations on how routes are constructed. We refer to Fig. 4 in the following paragraph. Using the *route proposal* sub-protocol described in Sect. 3.2, the routes were constructed in the following order:

- First, D proposed itself to its neighbor. Z learned here the value $LocalID_D^Z$, and D and Z formed the link $L_{Z,D}$.
- Then, Z proposed this newly learned route to its own neighbors. For each neighbor, a different link identifier was formed: $L_{Y,Z}$ and $L_{B,Z}$.
- In a third phase, B and Y simultaneously proposed their newly learned route to D. The new links are then $L_{X,Y}$ and $L_{A,B}$.
- After two more steps, S_1 and S_2 were aware of D.

Now, we can see that: (i) two sources may share a portion of route towards the same destination, and (ii) one source may have several routes towards the same destination.

For (i), there are two examples: S_1 and S_2 share the links $L_{Y,Z}$ and $L_{Z,D}$. In Fig. 4, the links for the green route and the blue dashed one are noted differently, but all links drawn between Y and Z are designated by the same link identifier $L_{Y,Z}$ (the same goes for links between Z and D). Therefore, whether a message was originally sent by S_1 or S_2, Z receives it from the same link identifier, and forwards it on the same link identifier. The second example is the portion of route between Z and D, shared between the dashed blue route coming from S_1 and the dotted one coming from S_2. But in this case, Z can make the difference between a message from S_1, which arrives from $L_{Y,Z}$, and a message from S_2, which arrives from $L_{B,Z}$.

Concerning the second fact (ii), we have an example with S_1. It has a route using X, and another using A. This means messages from the same communication between S_1 and D may take each a different path, and a corrupted node on one route will only see a fraction of the communication.

Property 2 *If the Decisional DH assumption holds and secure cryptographic key derivation functions exist, link identifiers are secure: for a given pair of neighboring nodes (X, Y), link messages are unlinkable for AdvI and AdvII. That is to say, the pairs* $(L^{(1)}_{\mathrm{X,Y}}, K^{(1)}_{\mathrm{X,Y}})$ *and* $(L^{(2)}_{\mathrm{X,Y}}, K^{(2)}_{\mathrm{X,Y}})$ *stemming from two different DH handshakes are unlinkable.*

As the link identifier structure is borrowed from a construction by Zhang *et al.* [22], we refer to their work. Note however that the authors, in order to achieve authentication during the neighbor discovery phase, make use of an external trusted third party and a bilinear map. As authentication is not a goal for us, we solely need regular Diffie-Hellman handshakes and cryptographic key derivation functions (KDF). Assuming the DH handshake led to the common value S, in our protocol, two neighboring nodes (X, Y) compute their i-th key and link identifier as $L_{X,Y} = \mathsf{KDF}(S, salt, 2i)$ and $K_{X,Y} = \mathsf{KDF}(S, salt, 2i+1)$. Therefore, if the Decisional DH assumption holds and KDF exists, the generation of link identifiers is secure in our sense. Indeed, the secret S can only be known by X and Y, thus they are the only ones able to generate such links. If the KDF is collision resistant, a different handshake or a different index i lead to different links and keys. If the KDF is one-way (or *pre-image resistant*), it is impossible for a PPT adversary to find v given $H(v)$, thus the secret S can not be found. Finally, if the KDF is pseudo-random, the link and key for index i and the ones for index $i+1$ both look random (to a PPT adversary), and are thus unlinkable. In the same way, the links and keys for two different DH handshakes are computationally indistinguishable from random, and are thus unlinkable (even for the entities that performed the DH handshake).

Property 3 *The local identifiers are secure: for all X and D,* $LocalID^{\mathrm{X}}_{\mathrm{D}}$ *is known only to X; if the Computational DH assumption holds, given* $LocalID^{\mathrm{X}}_{\mathrm{D}}$ *it is not possible to recover the inputs that formed it; and if the DDH assumption holds, for any X,* D_1, D_2, $LocalID^{X}_{D_1}$ *and* $LocalID^{X}_{D_2}$, *X is unable to know which local identifier points to which destination.*

Prior to formalizing the security definition of local identifiers, we abstract their structure. For this, we give an abstract definition of the computation of local identifiers in 3 steps, as in the route proposition. The first and last steps are made by the

destination, while the second is made by the source (or relay) node. We denote by $\mathsf{Gen} : \{0,1\}^* \to \{0,1\}^*$ the function taking in input the security parameter λ and generating the public parameters pp, including the sets $\mathcal{X}$, $\mathcal{Y}'$ and $\mathcal{Z}$. By $\mathsf{Comp1} : \mathcal{X} \to \mathcal{Y}$ we denote the ideal function computing the first step of the local identifiers computation on the *destination side*. $\mathsf{Comp2} : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$ is the ideal function computing the second step, on the *source side*. And $\mathsf{Comp3} : \mathcal{Y} \times \mathcal{Y} \to \mathcal{Z}$ is the ideal function computing the third step, on the *destination side*. We write $LocalID_D^X = \mathsf{Comp3}(y_D, \mathsf{Comp2}(x_X, \mathsf{Comp1}(x_D)))$, where $x_D \in \mathcal{X}$ is the input of X, and $x_D \in \mathcal{X}$ and $y_D \in \mathcal{Y}$ are the inputs of D. For simplicity, we define $\mathsf{Comp} : \mathcal{X} \times \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ as $\mathsf{Comp}(x_1, x_2, y) := \mathsf{Comp3}(y, \mathsf{Comp2}(x_2, \mathsf{Comp1}(x_1))$.

We say a local identifier structure is secure if the following properties hold for all nodes X and D:

- *Pairness*: $LocalID_{\mathrm{D}}^{\mathrm{X}}$ depends on values generated by X and D only. If $LocalID_{\mathrm{D}}^{\mathrm{X}} = \mathsf{Comp}(x,, x_2, y)$, then x_1, x_2 and y belong to one of X and D, and are secret to this node.
- *Secrecy*: $LocalID_{\mathrm{D}}^{\mathrm{X}}$ must be known by X only.
- *Collaborative oblivious computation*: For all X and for all D $\neq$ X, $LocalID_{\mathrm{D}}^{\mathrm{X}}$ can not be computed by a network entity by itself, and must be the result of a protocol between several parties where each party keeps its inputs secret.
- *Uniqueness*: For all nodes W, X, Y, Z such that $\neg(W = X = Y = Z)$, $LocalID_W^X \neq LocalID_Y^Z$ with high probability.
- *Indistinguishability*: Adversary $\mathcal{A}^{\mathcal{O}_{\mathsf{Gen}(\cdot)}, \mathcal{O}_{\mathsf{Comp}(\cdot,\cdot,\cdot)}} = (\mathcal{A}_1, \mathcal{A}_2)$ has negligible advantage in winning the game $\mathbf{Exp}^{\mathsf{ind}}_{LocalID}$ described below, that is to say the value $|Pr[\mathbf{Exp}^{\mathsf{ind}}_{LocalID}(\mathcal{A}^{\mathcal{O}_{\mathsf{Gen}(\cdot)}, \mathcal{O}_{\mathsf{Comb}(\cdot,\cdot,\cdot)}}) = 1] - 1/2|$ is negligible in the security parameter λ:

Game $\mathbf{Exp}^{\mathsf{ind}}_{LocalID}$ $(\mathcal{A})$

$pp \leftarrow \mathsf{Gen}(1^\lambda)$; $(x_1^{(1)}, y^{(1)}), (x_1^{(2)}, y^{(2)}) \hookleftarrow \mathcal{U}(\mathcal{X} \times \mathcal{Y})$
$(x_2, state) \leftarrow \mathcal{A}_1(pp, y^{(1)}, y^{(2)})$
$b^* \hookleftarrow \mathcal{U}(\{0,1\})$; $LocalID^* \leftarrow \mathsf{Comp}(x_1^{(b^*)}, x_2, y^{(b^*)})$
$b \leftarrow \mathcal{A}_2(state, LocalID^*)$
Return $b = b^*$

- *Computational One-Wayness*: For an adversary $\mathcal{A}$, the following probabilities are negligible in λ.

$Pr[pp \leftarrow \mathsf{Gen}(1^\lambda) : \mathcal{A}(1^\lambda, pp, x_2, \mathsf{Comp}(x_1, x_2, y)) = y]$

$Pr[pp \leftarrow \mathsf{Gen}(1^\lambda) : \mathcal{A}(1^\lambda, pp, x_2, y, \mathsf{Comp}(x_1, x_2, y)) = x_1]$

We now show that our structure of local identifiers indeed fulfills these properties. Note that in our case, x_1, x_2 and y respectively correspond to dst_D, src_X and ID_D, and that we want to prevent a node X who knows the value $LocalID_D^X$ from being able to link it to $y = ID_D$ or to recover the value $x_1 = dst_D$.

The *Pairness*, *Secrecy* and *Collaborative computation* properties relate to the dynamics of the network. We can only argument them. For the 3 other properties, proof sketches can be given. *Pairness* is trivially true: we use src_X which is generated and

belongs to X, and ID_D and dst_D which are generated and belong to D. *Secrecy* is indeed verified because $LocalID_D^X$ is never seen in clear by any other node than X: it is always encrypted, either by Elgamal either by link keys. Thus *Secrecy* relies on the security of those encryptions. The only exception appears in the route proposition, where D sees the values $r.g^{dst_D \cdot src_X}$ and $r.LocalID_D^X$. But D learns nothing from this value, because of the blinding factor r. *Collaborative oblivious computation* is also respected: local identifiers are indeed collaboratively computed in the route proposition and initialization. In the route proposition, src_X remains secret to X assuming the discrete logarithm is a hard problem, and dst_D and g^{dst_D} remain secret to D assuming Elgamal is semantically secure. As for ID_D, it remains secret if we manage to prove the *Computational one-wayness* property. In the route initialization, dst_D stays secret to D by the security of the THE scheme with respect to S, and thanks to the blinding factor r with respect to X. src_X stays secret by the security of Elgamal, and again ID_D is secret if *Computational one-wayness* holds.

Proof Sketch for Uniqueness. By reflexion on the birthday attack. It is know that, for a function $f : \{0,1\}^L \to \{0,1\}^l$ with $L > l$, and a collection of uniformly distributed elements $x_1, \ldots, x_k \in \{0,1\}^L$, the probability that at least one of the pair (x_i, x_j) with $i \neq j$ is a collision is roughly $p = \binom{k}{2} 2^{-l}$. In our case, $l = |q| \approx 2\lambda$, $L = 3l$, and k is the number of distinct local identifiers in the network. As there are n nodes, and each node has n local identifiers, $k = n^2$. For a constant maximum number of nodes equal to 10^6, the probability that two local identifiers are the same is:

$$p = \binom{10^{12}}{2} 2^{|q|}$$

This value is indeed negligible in the security parameters λ. For example, when $\lambda = 80$, $p \approx 2^{-80}$ and for $\lambda = 128$, $p \approx 2^{-180}$

Proof Sketch for Indistinguishability. As $LocalID_D^X = ID_D.g^{src_X \cdot dst_D}$ can be seen as an Elgamal encryption of ID_D under key dst_D and randomness scr_X, it would be convenient to borrow the semantic security proof of the scheme. However, it is not possible, because in our case the adversary has access to additional knowledge. Indeed, if the adversary is the node X, she knows src_X, which corresponds to the randomness used for encryption if we consider $LocalID_D^X$ as an Elgamal encryption. Hence, we need a different proof.

The idea is to show that, for a node X, given src_X of its choice, the identity of two nodes ID_{D_1} and ID_{D_2}, and the value $LocalID_{D_*}^X$ (its local identifier for one of D_1 or D_2), X can not know if $LocalID_{D_*}^X$ is equal to $LocalID_{D_1}^X$ or $LocalID_{D_2}^X$. Let's review the possibilities of X. She can proceed as follows: it begins by assuming the local identifier is towards D_1 and computes (assuming src_X and $p-1$ are co-prime):

$$\sqrt[src_X]{\frac{LocalID_{D_*}^X}{ID_{D_1}}} = g^{dst_{D_*}} \left(\frac{ID_{D_*}}{ID_{D_1}}\right)^{1/src_X}$$
$$= g^{dst_{D_*}}.g^{(logID_{D_*} - logID_{D_1})/src_X}$$

In both cases, what X obtains is computationally indistinguishable from a uniformly sampled element of $\mathbb{G}$. Indeed, if $D_* = D_1$, X obtains $g^{dst_{D_1}}$ which is uniformly distributed, because dst_{D_1} is itself uniformly distributed (and unknown to X). And if $D_* = D_2$, even though X can compute $g^{(logID_{D_*} - logID_{D_1})/src_X}$ and obtain $g^{dst_{D_2}}$ the

result is again uniformly distributed for the same reason. If X had assumed that D_* was D_2, the same argument applies.

The other possibility for X is to first divide by ID_{D_1} and compute the root afterwards. But the result is the same, and even though X may find $g^{dst_{D_*}}$ in several ways, it can never confirm or refute its hypothesis and distinguish the two cases. So $\mathcal{A}$ can not win the game with non-negligible probability.

Proof Sketch for Computation one-wayness. This property follows from the *Indistinguishability* property and from the hardness assumption of the discrete logarithm problem. Indeed, if we consider the first probability. If the adversary was able to recover y with with non-negligible probability, she would win the game $\mathbf{Exp}^{\mathsf{ind}}_{LocalID}$ with non-negligible probability (assuming $Pr[y^{(1)} = y^{(2)}]$ is negligible, which is the case). Regarding the second probability, the adversary's success is bounded by its ability to compute a discrete logarithm.

Property 4 *Route propositions are indistinguishable for AdvI. Because, locally, the messages exchanged are the same in both cases, whether a route proposition emanated from the destination itself or a proxy is indistinguishable for AdvI.*

This property can also be modeled by a cryptographic game. We consider a setup where either the node X is a direct neighbor to D, either there are one or several nodes between X and D. We give the adversary the right to provoke at will as many propositions as it wants, but not to change the network setting after the beginning of the game. The goal of the adversary, of type AdvI and in the situation of X, is to distinguish the two cases (finding out if the destination D of the proposition is a direct neighbor or not). We assume that the adversary can not perform timing analysis. We discuss this assumption in the paragraph concerning property 5.

Proof sketch. Informally, it is easy to show that from the point of view of X, and without taking in account message transmission delays, the two cases are exactly the same. Indeed, the number of messages received and emitted by X is respectively two and one in both cases. Also, the respective size of the first, second and third messages are the same in both cases. The first message contains the destination's public key, a half of DH handshake freshly (uniformly) generated, and a re-randomized ciphertext always encrypting the same value. Assuming the re-randomization indeed outputs ciphertexts uniformly distributed in the range of the encryption primitive, the distributions of the first message in each case are computationally indistinguishable. The second message is produced by X, and thus is not of interest here. Then, the third message contains a link identifier and a symmetric encryption of $r.LocalID^X_D$ using the link key. The distributions of the third message are thus also computationally indistinguishable, assuming link identifiers are secure, and that the correct value $r.LocalID^X_D$ is always returned.

For a formal proof, it would be necessary to simulate and compare X's views in the two cases.

Property 5 *If route propositions are indistinguishable for AdvI, if the encryption schemes used are semantically secure, if the link identifiers are secure, and if we use MIXing techniques in the route proposal, then up to a certain point, AdvII can not distinguish whether a proposition emanated from the destination or a proxy.*

Firstly, if AdvII does not perform traffic analysis but merely records the message and only has a local view, she is not able to distinguish whether a route proposition

emanated from the destination itself or a proxy. By property 4, the respective distributions of the first and third message are the same whether X is a direct neighbor to D or not. And by the semantic security of the PKE scheme used, the re-randomization property, and the uniform sampling of b (and thus g^b), the distributions of second message are also computationally indistinguishable. Now, if AdvII is able to perform traffic analysis and has a global view, she can trace sequences of route proposals moving away from the destination D. Thus, AdvII can distinguish when the route is proposed by the destination itself or not. To thwart this, we can use message batching and reordering. As a result, AdvII is no longer able to follow sequences of route proposals. However, what we can *not* prevent are timing analysis. For a proxy at, say, 1 or 2 hops from D, thanks to random delays, AdvII will have difficulties distinguishing the two cases using differences in timings, as they will be small. However, the time required to go from X to D and come back is much more important in the case X is at 4 or more hops from D than in the case X is a direct neighbor. Thus, AdvII can not know if a proposition emanates from a destination or a proxy, up to a certain number of hops.

Property 6 *If local identifiers are secure, route initialization is oblivious for AdvI. Indeed, S never makes the link between D and its value* $LocalID^{\mathrm{S}}_{\mathrm{D}}$*, and V never learns that the destination is D. In addition, the blinding factor* r *hides* dst_{D} *and if the discrete logarithm problem is hard,* src_{X} *is concealed.*

In the same way as for the local identifiers, we formalize the security requirements for route initialization. We say route initialization is secure if the following properties hold:

- *Obliviousness for S*: S must not be able to link ID_D with $LocalID^S_D$.
- *Obliviousness for X*: X must not learn the destination that S considers.
- *Secrecy*: each party N should keep its values src_N, dst_N, ID_N and all the local identifiers in its DRT secret to himself. D should keep its value dst_D secret.

The two first properties are necessary for DMU to hold, while the third one ensures the assumptions on the secret values src, dst and ID holds, and that the *Secrecy* of local identifiers is conserved. We now prove that our route initialization fulfills these properties.

Proof sketch for Obliviousness for S. We assume that the local identifiers are secure, in the sense of property 3. Thus S can not *a priori* link ID_D (or simply D) with $LocalID^S_D$ because of the *Indistinguishability* property of local identifiers. Then, the route initialization procedure do not give S any new element towards linking ID_D. Indeed, if the THE scheme is semantically secure, S does not learn anything from the values $C = \mathsf{THEnc}(pk_*, dst_D)$ and $\mathsf{THDec}(sh_S, C)$. And if Elgamal is semantically secure, S does not learn anything from $\mathsf{HEnc}(pk_X, g^{r.dst_D.src_X})$. At last, because we assumed that every node has a route towards every other node in the network, the fact that X knows a route towards D does not leak information.

For a formal proof, it would be necessary to simulate S's view during the route initialization.

Proof sketch for Obliviousness for X. We assume that the local identifiers are secure, in the sense of property 3. Thus X can not *a priori* know that its value $LocalID^X_D$ links to D. Then, thanks to the blinding factor r that acts as a one-time pad, X does not learn anything from the decryption of $C = \mathsf{THEnc}(pk_*, r.dst_D)$. Then, if the

re-randomization yields random-looking ciphertexts, X can not extract information (in particular on r) from $\mathsf{ReRand}(pk_{\mathrm{X}}, \mathsf{ScMult}(ID_{\mathrm{D}}, \mathsf{ScExp}(C', r^{-1} \mod (p-1))))$. X eventually learns $LocalID_D^X$, but nothing else, and compared to the *a priori* case, X has no more information on the link between ID_D and $LocalID_D^X$. Finally, although it is not relevant in this proof, it is worth noting that if the secret share sh_S is not publicly linked to S, even if the value $\mathsf{THDec}(sh_S, C)$ leaks information on sh_S, X can not infer that the source is S.

For a formal proof, it would be necessary to simulate X's view during the route initialization.

Proof sketch for Secrecy. During the protocol, S keeps its values src_S, dst_S and ID_S, and its local identifiers secret simply because it does not use them. The same goes for X's values dst_X and ID_X, and all local identifiers of X (except $LocalID_D^X$). Then, the values src_X, g^{src_X} and $LocalID_D^X$ are not disclosed assuming Elgamal is semantically secure. Finally, as we consider *honest-but-curious* adversaries, S and X do not collude and S indeed applies a scalar multiplication by a blinding factor r on $\mathsf{THEnc}(pk_*, dst_D)$ (and a re-randomization). Therefore, dst_D is ensured to stay secret to D.

For a formal proof, it would be necessary to simulate the view of S and X during the route initialization.

Property 7 *If we use MIXing techniques, and if link identifiers are secure, route initialization is indistinguishable from standard communication for AdvII.*

This property is quite trivial. Messages exchanged between S and X in the route initialization are encrypted with link keys and addressed *via* link identifiers, just as in a regular communication. Therefore, if the link identifiers are secure in the sense of property 2, an external observer is unable to see the difference between route initialization and regular communication. The only possibility for the observer is to *fingerprint* the route initialization protocol by simulating runs of this protocol and measuring average timings between messages or other metrics. Then, the observer could possibly identify routes initialization in the network, because the sequence of messages exchanged is different from standard communication in general. We introduce MIXing techniques in the route initialization, so that obtaining a precise fingerprint is difficult for the adversary.

B.3 Privacy goals

This section shows that our protocol achieves our actual privacy goals. From now on, we assume that the seven properties from the previous section are ensured. Also, as a general remark, note that AdvII has very little power thanks to the link identifier structure (assuming they are secure). Indeed, all messages are encrypted and locally addressed with link identifiers in an anonymous way. Furthermore, for a given message, its appearance changes at each hop, and for a given link, the symmetric encryption key changes at each message. And also, because we assume messages are of fixed size, AdvII can not learn anything on the contents of messages and can not trace them.

SMU against AdvI This property is quite simply verified. Locally, when a node receives a message on some link, it can not know if the node at the other end of the link acts as source or relay (notably because all nodes may be source or relay). Furthermore, the contents of the messages do not give any information on the source, either directly (*e.g.* its identity) or indirectly (*e.g.* the number of hops from the source).

SMU against AdvII If property 7 holds, AdvII can not detect a route initialization, and she can not know which nodes initiate communications by this mean. However, it is not sufficient. Indeed, using the simple heuristic that a node emitting a message without prior solicitation is a sender, AdvII can locate sources (but not learn their identities). Even though the chances of locating sources are reduced when traffic is dense and when MIXing techniques are employed, if AdvII notices that a node receives k messages in some time frame and emits $k+1$ messages afterwards, she will infer that the node is a source. The feasibility of this attack is however uncertain. Especially in broadcast networks such as ad-hoc networks (or more generally, wireless networks): if each node locally broadcasts its messages (either as a source or as a relay) and if we make use of secure link identifiers to address neighbors, AdvII can not count the number of messages received by a given node. Therefore, she can not compare the number of incoming and outgoing messages from a node nor locate sources.

DMU against AdvI First, if local identifiers are secure (property 3), they do not reveal the destination they are pointing to. Secondly, if property 4 holds, route propositions do not let the receiver of the proposition know who the actual destination of the proposition is. Then, if property 6 holds, route initialization does not leak information to X about the destination intended by S, and S does not learn that $LocalID_D^S$ relates to D. At last, it is necessary to assume that the PKE scheme used to encrypt payload messages is *receiver-anonymous* [15]. Indeed, the permanent public keys are publicly linked to their owners, and in general, ciphertexts of PKE schemes leak information on the encryption key. Thus, forwarding nodes may learn the identity of the destination. In receiver-anonymous PKE schemes, it is ensured that no information is leaked by ciphertexts on the encryption key.

DMU against AdvII Firstly, if property 5 holds, AdvII can not learn the destination of a route proposition by observing the route proposition protocol. Then, if link identifiers are secure, AdvII can not trace messages to their destination. However, in the same way that she locates sources, AdvII can locate destination. To thwart destination localization, destination nodes can simulate a relay upon receiving a message. A possible way to do so, is for a destination to create a fake message containing solely a TTL-like number and padding. The TTL may take a value between 0 and 2 for instance, randomly chosen. The destination sends this value on a random route chosen in its DRT. The addressed neighbor, upon receiving the fake message, if the TTL value is greater than 0, decreases it by 1 and forwards it on a random route as well. When the TTL reaches 0, the message is discarded. As a result, to the knowledge of AdvII, the possible receivers of a given message are the node that received k messages in some time frame but emitted $k-1$ messages afterwards, along with its 3 hop neighborhood.

SDU against AdvI and AdvII This property is quite straightforward: if SMU holds against some adversary, then the source of a communication is unknown and SDU holds. The same goes for DMU. Therefore, if either SMU or DMU hold, SDU holds.

MMU against AdvI This follows from property 1 stating that routes are disjoint. Indeed, in Fig. 4, a locally corrupted Z is actually completely unable to distinguish

a message from S_1 and a message from S_2. Because it only has a local view of the network, given an incoming link identifier, Z does not know if this link is used by one upstream source only or by several ones. Therefore, even though Z may know that two messages are meant for the same destination, it does not know if it comes from the same source, and can not infer that those messages belong to the same communication.

However, in the route initialization example from Fig. 3, an adversary of type I who corrupted X is able to break MMU. Indeed, it knows that all messages come from the same source and are for the same destination. This vulnerability is acceptable because X does not know the identity of S and D.

MMU against AdvII For this property to hold, link identifiers must be secure (property 2). A noted above, AdvII is unable to learn any information on the contents of messages, can not trace them, and thus can not relate two messages. The only information AdvII can infer is that two messages going though the same link *may* belong to the same communication. But she can not be sure of it, thanks to property 1.

B.4 Remark

Our argumentations does not consider the auxiliary information that the adversary may have. For instance, the adversary might know that some node is a privileged destination, maybe because it is the only one routing messages to Internet. As AdvII can detect traffic patterns, she will easily locate this node: it will be the target of a considerable amount of traffic. There are many other examples of the same sort. Taking into account the adversary's auxiliary information is part of future work.

C (Imp)possibilities for route initialization

In Sect. 4.1, we discussed the necessity of *universal auxiliary information* to enable *secure* and *efficient* route initialization. This appendix explicits what we mean by "secure and efficient", and why, in our model, our claim is true. As starting point, we assume the topology is disseminated and consider the local identifiers in their abstract description from property 3 in Appendix B, assuming they are secure. In other words, this section does not consider any particular instantiation for the local identifier structure.

We already addressed the notion of *security* for route initialization in property 6 from Appendix B. We measure *efficiency* according to the number of nodes in the network. As we will see in this appendix, there is a (relatively) trivial way for S and X (in our example in Fig. 3) to *find* the value $LocalID^{\mathrm{X}}_{\mathrm{D}}$ in X's DRT. The idea is to make a sort of exhaustive search in both S and X's DRTs. The cost of this solution in term of computation and communication is quadratic in the number of nodes n in the network: its complexity is $O(n^2)$. We consider reducing this complexity: we are interested in solutions which cost is *linear* in n or even *constant*, *i.e.* in $O(n)$ or $O(1)$ with respect to the number of nodes.

According to our results, there exists solutions with such complexities, under particular assumptions. The below claims and their proof sketches capture our results.

Claim 1. *A secure solution for route initialization in $O(n^2)$ is possible (against a PPT honest but curious adversary).*

Proof Sketch. The proof consists in exhibiting a route initialization protocol with cost $O(n^2)$, and to show it is secure (according to our definition). A possible, highly inefficient solution is depicted in sub-protocol 2, where "$\forall$ N $\in$ $\mathrm{DRT_X}$" denotes all the entries of X's DRT for all destinations N. Clearly, the cost of this solution is linear in the number of different local identifiers in the DRT of S, and linear in the number those in the DRT of X. That is to say, it is quadratic in n.

The *correctness* of the protocol is easily checked: the value $test_{N,M}$ is indeed equal to 1 if N = M = D. In every other case, the probability that the plaintext in equation (2) is equal to 1 is roughly $1/|\mathbb{G}|$. Now, we argue that the security properties of route initialization hold for this sub-protocol.

Globally, the *obliviousness for S*, the *obliviousness for X* and the *secrecy* properties rely on the semantic security of the HE scheme used. Indeed, both S and X mainly manipulate ciphertexts. There are however exceptions that might compromise security:

- X sees in the clear the values $(LocalID_M^X)^{src_S}$ for all M. But it learns nothing new from them if the discrete logarithm is a hard problem.
- S sees in the clear the values $text_{N,M}$ for all N and M. However, because the identities and the *src* and *dst* values are randomly distributed, and because X processes the $aux_2(S, M)$ and $aux_2(X, N)$ in random order, the values $test_{N,M}$ are indistinguishable from random for S.

This concludes the argument. Proofs by simulation using standard hybrid arguments (one for S, one for X) are necessary to formally prove our claim.

Claim 2. *A secure solution for route initialization in $O(1)$, without universal auxiliary information, is impossible (against a PPT honest but curious adversary).*

Proof Sketch. By contradiction. Imagine that a *secure* and *correct* protocol $\mathcal{P}$ exists for a route initialization in $O(1)$, without using universal auxiliary information.

In a solution with complexity $O(1)$ (w.r.t. n), we forbid the scanning of both S and X's DRTs. Therefore, the respective inputs of the protocol can be described as:

- S's input contains $x_{1,S}, x_{2,S}, y_S$. It can contain: dynamically generated values such as cryptographic keys, and auxiliary values related to D (because we assume D communicated information to S prior to network setup). It can *not* contain: universal auxiliary information on the network.
- X's input contains: $x_{1,X}, x_{2,X}, y_X$. It can contain: dynamically generated values such as cryptographic keys. It can *not* contain: universal auxiliary information on the network, nor any auxiliary information on D (indeed, X does not know *a priori* which node is S's intended destination).

We denote the union of S and X's inputs by $In_S \cup In_X$. Now, there are two possibilities: either $In_S \cup In_X$ contains the necessary material to compute $LocalID_D^X$ using protocol $\mathcal{P}$, either it does not. In the latter case, we reach a contradiction because we supposed that $\mathcal{P}$ was *correct*. And if $In_S \cup In_X$ allows computation of $LocalID_D^X$, then we also reach a contradiction: S can compute by itself $LocalID_D^S$, thus making the link between D and $LocalID_D^S$ and breaking the *Obliviousness for S* property of route initialization (and, ultimately, S breaks DMU as explained in Sect. 4.1). Indeed, if $In_S \cup In_X$ can lead to $LocalID_D^X$, this means that $x_{1,D}, x_{2,X}, y_D \in In_S \cup In_X$, because $LocalID_D^X = \mathsf{Comp3}(y_D, \mathsf{Comp2}(x_{2,X}, \mathsf{Comp1}(x_{1,D})))$. However, X's inputs can not contain information on D. This implies that $x_{1,D}$ and y_D come from S's inputs. Therefore, in S's inputs, there is the necessary material to compute $LocalID_D^S = \mathsf{Comp3}(y_D, \mathsf{Comp2}(x_{2,S}, \mathsf{Comp1}(x_{1,D})))$. Said otherwise, S can run the protocol $\mathcal{P}$ playing both roles

in order to obtain $LocalID_D^S$. This is because X's input, with respect to $LocalID_D^X$, does not contain anything more than S's input with respect to $LocalID_D^S$.

Claim 3. *There exists a secure solution for route initialization in $O(1)$, assuming the existence of universal auxiliary information (against a PPT honest but curious adversary).*

Proof Sketch. To prove this claim, it is sufficient to exhibit a particular solution for route initialization in $O(1)$ that uses universal auxiliary information and is secure. We already provided such a solution in the description of our protocol: the route initialization described in Sect 3.3 indeed uses universal auxiliary information (under the form of secret shares of sk_*) and runs in constant time with respect to the number of nodes in the network (indeed, no DRT or any table is used). Therefore, this claim follows directly from the proof sketch of property 6 in Appendix B.

In the above claims, we did not address the case of solutions in $O(n)$, linear in the number of nodes. Actually, it seems that the same claims as for the constant case hold: a solution in $O(n)$ is possible only using universal auxiliary information. Indeed, we reach the same case, where S is able to run the protocol playing both roles in order to obtain $LocalID_D^S$. We did not further study solutions in complexity $O(n)$ because their interest is limited. Indeed, they seem to yield less efficient solutions for the same strong assumption as the solutions in complexity $O(1)$.

Protocol 2 S initiates a connection towards D with help of X in $O(n^2)$

1: $S \to X : \mathsf{HEnc}(pk_S, ID_D), \mathsf{HEnc}(pk_S, ID_D^{src_S}), \forall N \in DRT_S, \mathsf{HEnc}(pk_S, LocalID_N^S)$
2: $X \to S : \forall M \in DRT_X, \mathsf{HEnc}(pk_X, LocalID_M^X)$

For all M, S computes $\mathsf{ScExp}(\mathsf{HEnc}(pk_X, LocalID_M^X), src_S)$
$= \mathsf{HEnc}(pk_X, ID_M^{src_S}.g^{src_X.dst_M.src_S})$

3: $S \to X : \forall M, \mathsf{HEnc}(pk_X, ID_M^{src_S}.g^{src_X.dst_M.src_S})$ (in the same order as received)

For all M, X decrypts the value received and re-encrypts it under pk_S
(This step is necessary to perform the following homomorphic operations on ciphertexts).
X computes $aux_1(X) = \mathsf{ScExp}(\mathsf{HEnc}(pk_S, ID_D), src_X) = \mathsf{HEnc}(pk_S, ID_D^{src_X})$
Denote by $aux_1(S)$ the value $\mathsf{HEnc}(pk_S, ID_D^{src_S})$ from message 1
For all N, X computes $aux_2(X, N) = \mathsf{ScExp}(\mathsf{HEnc}(pk_S, LocalID_N^S), src_X)$
$= \mathsf{HEnc}(pk_S, ID_N^{src_X}.g^{src_S.dst_N.src_X})$
Denote by $aux_2(S, M)$ the values from message 3

4: While X did not receive STOP from S, for all M, **in random order**, X computes and sends, for all N:

$$test_{N,M} = \frac{aux_1(S).aux_2(X,N)}{aux_1(X).aux_2(S,M)} \quad (1)$$

$$= \mathsf{HEnc}\left(pk_S, \frac{ID_N^{src_X}}{ID_M^{src_S}} \cdot \frac{ID_D^{src_S}}{ID_D^{src_X}}.g^{(dst_N - dst_M).src_X.src_S}\right) \quad (2)$$

$$= 1 \text{ if N = M = D, and a random value otherwise} \quad (3)$$

$X \to S : \mathsf{HEnc}(pk_S, test_{N,M})$

5: $S \to X : \mathsf{STOP}$ if $test_{N,M} = 1$
X infers that $LocalID_M^X$ is the local identifier sought for.