Pers Ubiquit Comput
DOI 10.1007/s00779-012-0619-x

ORIGINAL ARTICLE

An efficient privacy-preserving solution for finding the nearest doctor

George Drosatos · Pavlos S. Efraimidis

Received: 30 September 2011 / Accepted: 17 July 2012

Abstract In this work, we define the Nearest Doctor Problem for finding the nearest doctor in case of an emergency and present a privacy-preserving protocol for solving it. The solution is based on cryptographic primitives and makes use of the current location of each participating doctor. The protocol is efficient and protects the privacy of the doctors' locations. A prototype implementing the proposed solution for a community of doctors that use mobile devices to obtain their current location is presented. The prototype is evaluated on experimental communities with up to several hundred doctor agents.

Keywords Location privacy · Personal data · Privacy-preserving computation · Peer-to-Peer network

1 Introduction

The advances in information and communication technologies (ICT) and the wide acceptance of electronic transactions for everyday tasks of individuals have a strong impact on the use and protection of personal information. Desktop and mobile computing technology, sensors and the advances in database and storage technologies have increased the amount of personal information that is generated and the potential for this information to be (permanently) stored and processed. Any kind of personal information that results as an outcome of electronic activities of individuals, either personal or professional, belongs to the category of personal data. Personal data is a critical, valuable resource that has to be protected in order to ensure the individual's privacy rights.

The same advances in ICT that cause the generation of more personal information also provide a vast potential for emerging new applications that can use personal data in favor of the individuals' interests. Some examples are personalized web services that automatically adapt to the profile of an individual, and location-based services that behave according to the individual's current location or context. Individuals, as well as the society as a whole, may obtain significant benefits if personal data can be used legitimately for rightful purposes. However, the use of personal data should be done in a way that simultaneously ensures their protection. Each individual has the right to protect his privacy by retaining the control over his personal data and knowing who, when and why gets access to his data. Furthermore, when an individual makes a transaction, only the minimum possible amount of personal information that is needed to complete it should be disclosed. That is, release of personal data should be done in such a way that only the absolutely necessary items are disclosed and only when it is really needed. Moreover, the disclosure should take place with clear terms on how the personal data will be used.

A very interesting class of personal data is dynamic personal data, such as the current location of an individual. The recent progress in mobile device technology and the advances in ubiquitous computing allow individuals to collect and process such dynamic personal data. This enables a

A preliminary version of this work has been published in 3rd International Conference on PErvasive Technologies Related to Assistive Environments (PETRA 2010).

G. Drosatos (✉) · P. S. Efraimidis
Department of Electrical and Computer Engineering, Democritus University of Thrace, GR671 00 Xanthi, Greece
e-mail: gdrosato@ee.duth.gr

P. S. Efraimidis
e-mail: pefraimi@ee.duth.gr

Published online: 30 October 2012

new class of important applications. Consider, for example, the location of an expert, and in particular a doctor. In case of an emergency, the distance of the closest doctor could be live-saving information. In August 2007, in the area of Alexandroupolis, Greece, a 17-year-old boy was seriously injured in his right leg. Vascular surgery was urgently needed. However, due to several administrative faults, no specialized doctor was available. Even worse, it took a long time until it became clear that no specialized doctor could be found, and only then the boy was transported to a hospital in Thessaloniki. Unfortunately, due to the long delay, the injured leg had to be amputated. Even with the transport taking place, had the initial delay to find out where the nearest specialized doctor is been avoided, the consequences on the boy's health might have been less serious [34].

In this paper, we focus on dynamic personal data and examine the possibility of development of innovative applications that exploit this kind of data, while ensuring the privacy of individuals. To this end, we propose the following problem, called the Nearest Doctor Problem (NDP), to find the nearest doctor in case of an emergency. In a hypothetical but feasible scenario, each doctor has a personal agent where his current location is always stored. In case of an emergency, the agents of all doctors interact to identify the doctor who happens to be closer than any other to the emergency location. We assume that the doctors may be off duty, and thus, the current location of each doctor is sensitive personal data that should not be revealed to anyone, including other doctors.

The NDP is an example of a privacy-preserving application based on dynamic personal data, the location of each individual doctor. We propose a privacy-preserving solution that solves the problem without revealing the location of any doctor. The individual who is anonymously identified as the closest doctor can then reveal his identity and offer his services to the emergency event. The privacy guarantees of this work concern the current location of each doctor, which is the only personal information of the participants (doctors) used in the computation. Our approach solves the NDP without any doctor location being disclosed; only a small amount of aggregate or anonymous information about participants distances is leaked.

The solution that we propose for NDP makes use of cryptographic primitives and decentralized computation technologies. A basic assumption is that all doctors have at their disposal a personal data management agent where their current location is stored. Each agent is under the control of its owner, and all personal agents are permanently connected to the Internet. We consider this assumption feasible because, on the one hand, most modern smartphones (even low cost smartphones) are equipped with a Global Positioning System (GPS) which allows the users to monitor their current location and, on the other hand, the doctor agent can exist in a cheep, low-energy consuming nettop or possibly even in the Asymmetric Digital Subscriber Line (ADSL) router.

In case of an emergency, the agents of all doctors execute a distributed computation to identify in a cryptographic safe way who is the closest doctor to the incident. For performance, scalability and fault tolerance reasons and additionally for enhancing privacy the computation are executed in a fully decentralized way. The agents/nodes are (self-)organized in a distributed topology. To achieve this, we employ techniques from the field of Peer-to-Peer (P2P) networks. The use of P2P techniques allows us to satisfy the requirements of high scalability of the system and to reduce the risk for privacy breaches. We apply techniques that have been developed by Stamatelatos et al. [32] and are based on the well-known Chord [33] architecture for P2P networks.

The NDP is an illustrative example of an application where personal data can be used for a common good (public health), whereas at the same time the privacy of all involved individuals is preserved. We believe that many new applications can emerge from the same principle of simultaneously using and protecting personal data. For example:

- First aid in case of a car emergency. The European Union has launched the eCall project [13] for dealing with the ability of providing assistance in case of car emergencies. The project's goal is to deploy a hardware black box installed in vehicles that will send an emergency request in case of an accident on the road. The request will be transmitted over wireless communication technologies like the Global System for Mobile Communication (GSM) and will include information like the GPS coordinates of the emergency location, airbag deployment and impact sensor information. An additional action could be to search whether anyone in the nearby cars could offer first aid (who would be entitled to offer help in such cases is an issue that is out of the scope of this work). However, the location of a vehicle is private information, and so the search for nearby cars has to be done in a privacy-preserving way. An approach like the NDP solution presented in this work could be used to identify a nearby car. A different problem, probably easier than NDP, would be to warn all nearby cars to slow down.
- Police or fire emergency. In case of a police or fire department emergency, a policeman or a fireguard who is not on duty and happens to be near the event location might be able to provide critical services if he is informed about the emergency. At the same time, since the individual (policeman or fireguard) is not on duty, the exact location of a person is sensitive personal data

and nobody has the right to know it. A solution like NDP could identify such an individual (with his consent). The individual would be contacted by his own agent only if he is the closest person and if he is close enough to be able to help in such an emergency.

Related work The Active Badge Location System [35] was the first indoor location system for contacting people in an office environment. The system raised issues on location privacy at work. Extensions of the initial system and follow-up projects [36] offered enhanced features to the users for controlling the way their location data are accessed. However, all these systems assume a trusted server that manages the location data. A system that assumes a decentralized control of personal data is the Cricket Location-Support System [27]. Cricket describes an approach that offers an individual the option to learn his physical location within a building (that offer the Cricket service). The user can then decide to whom he discloses his location. This approach offers a better control over who obtains the location information of the individual. However, if the user wants to actively use his location information to perform some task, he has to disclose it. An approach like Cricket could be used to allow individuals to learn their location when they are within buildings where GPS cannot be used. All the above location systems are for indoor applications.

The privacy concerns for applications like NDP are even more critical since they apply to individuals who may be in their private time and not only at their office but at any location. Therefore, we present a privacy-preserving solution for NDP. The solution is based on secure multi-party computations (MPCs), that is, computations that receive input from two or more parties and calculate the output without revealing the input of any participant. General models for MPC have been proposed in the seminal work of Yao [37] and in follow-up works. However, the general models are practically inefficient. More efficient approaches are being developed for specific applications like, for example, the approach by Yokoo and Suzuki [2, 38]. A first large-scale and practical application of multi-party computation took place in Denmark in January 2008 [3]. A centralized approach using four separate servers was used to implement an electronic double auction that enabled Danish farmers to trade contracts for sugar beet production on a nationwide market. The NDP solution presented in this work is a decentralized, efficient privacy-preserving scheme for the NDP.

Outline The rest of this paper is organized as follows. In Sect. 2, we define the NDP. In Sect. 3, we present in detail the proposed solution for the NDP and propose an effective network topology in which the NDP protocol can be executed. In Sect. 4, we discuss the protocol's security under the Honest-But-Curious and Malicious models. The prototype implementation is presented and evaluated in Sect. 5. In Sect. 6, we discuss problems and issues related to the acceptance and the employment of a solution like NDP. Finally, conclusions of this work are given in Sect. 7.

2 The NDP problem

In this section we define the Nearest Doctor Problem (NDP). The main goal of NDP is to find the nearest doctor without violating the privacy of doctors. The personal data which are needed for the NDP computation are the exact locations of all doctors. An instance of the NDP consists of the following:

- **N doctors** $D_1, D_2, \ldots, D_N$.
- For $i = 1, 2, \ldots, N$, let L_i be the current location of doctor D_i. For instance, the location L_i may be the exact GPS location of the doctor, obtained from a portable GPS device.
- **The NDP lookup function:** In case of an emergency, (the agents of) all doctors perform a distributed privacy-preserving computation.
  - **Input:** The location L_{em} of an emergency.
  - **Output:** At the end of the computation, the doctor who is the nearest one to the location of the emergency becomes aware of this fact and can offer his services.

3 A solution for NDP

We describe a distributed privacy-preserving computation for solving the NDP. An overview of the architecture of the solution is presented in Fig. 1. The communication between entities in our architecture is performed over secure sockets (SSL/TLS) with both server and client authentication enabled. At the heart of our approach is a cryptographic protocol for a secure distributed computation.

3.1 Assumptions

We make the following plausible assumptions:

- Every doctor has a personal data management agent with permanent access to the Internet.
- The current location of each doctor is known by his or her personal agent.

3.2 Security model and privacy

We first define the security model and the kind of privacy that is achieved and then proceed with the description of

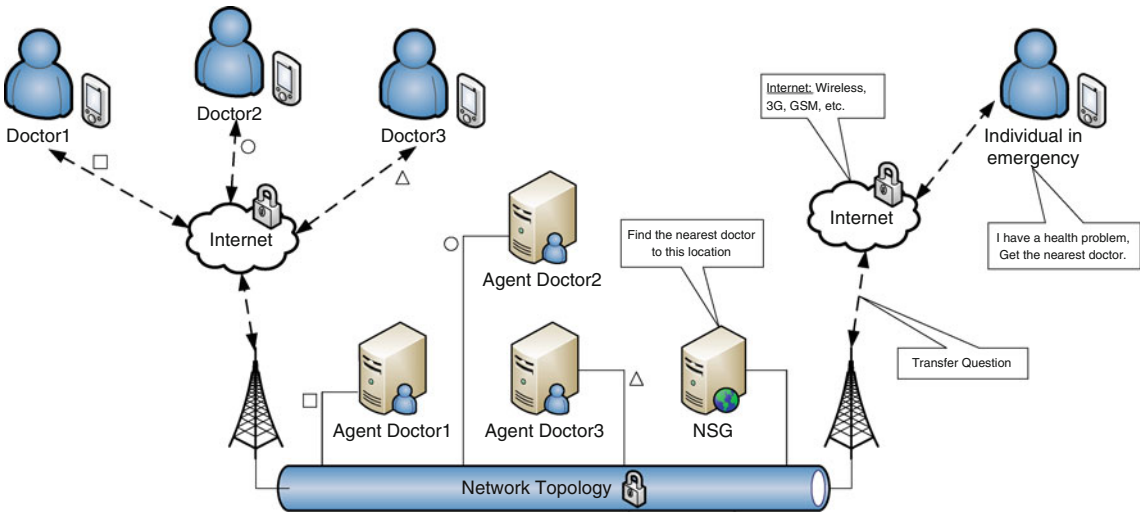


Fig. 1 The architecture of NDP solution

the distributed computation in the next section. We will show that the proposed protocol is safe in the Honest-But-Curious (HBC) model, that is, the doctors are assumed to follow the protocol steps but also may try to extract additional information (see Definition 3). The HBC model is commonly used in cryptographic protocols and is well suited for the NDP, since the participants are certified doctors. In Sect. 4.2 we go beyond the HBC model and examine how to handle some cases of malicious user behavior.

Regarding privacy, there are two distinct problems that arise in the setting of privacy-preserving computations [23]:

- The first is to decide which functions can be safely computed, where safety means that the privacy of the participants is preserved if the result of the computation is disclosed. We will assume that the outcomes of the distributed computations do not violate the privacy of the participating doctors and will not further consider this problem in this work.
- The second is how, meaning with which algorithms and protocols, to compute the results while minimizing the damage to privacy. For example, it is always possible to pool all of the location data in one place and run the computation algorithm on the pooled data. However, this is exactly what we do not want to do; doctors, and any individual in general, would not agree to continuously disclose their location data. The focus of our work is exactly on this problem.

Thus, the question we address is how to identify the nearest doctor without pooling the location data, and in a way that reveals (almost) nothing else about the distributed computation.

3.3 The NDP service

At the application level, the NDP solution is offered as a service through a dedicated node, called the NDP Service Gateway (NSG). When an individual is in an emergency, the following steps take place (Fig. 2):

- The individual or some authority submits a request with the emergency incident to the NDP Service Gateway (NSG). The request contains the current location of the individual (e.g., the exact geographic coordinates) and possibly additional information about his identity, his current condition, etc. Note that in the NDP, the location of the emergency is not considered private.
- The NSG is an access point that accepts the request and forwards it to an agent of the doctor's community. The agent which receives the request from the NSG takes the role of the root node for the particular computation.
- The root node coordinates a distributed computation that calculates the distance of the nearest doctor.
- At the end of the distributed computation, the agent of the doctor who is the nearest to the location of the emergency becomes aware of this fact and contacts the NSG to declare his readiness to offer help.

3.4 Outline of the distributed computation

We present a protocol for a secure distributed computation that solves the NDP. The protocol does not disclose the location of any doctor; only a small amount of aggregate or anonymous information is leaked. The computation consists of three main phases (Fig. 2).

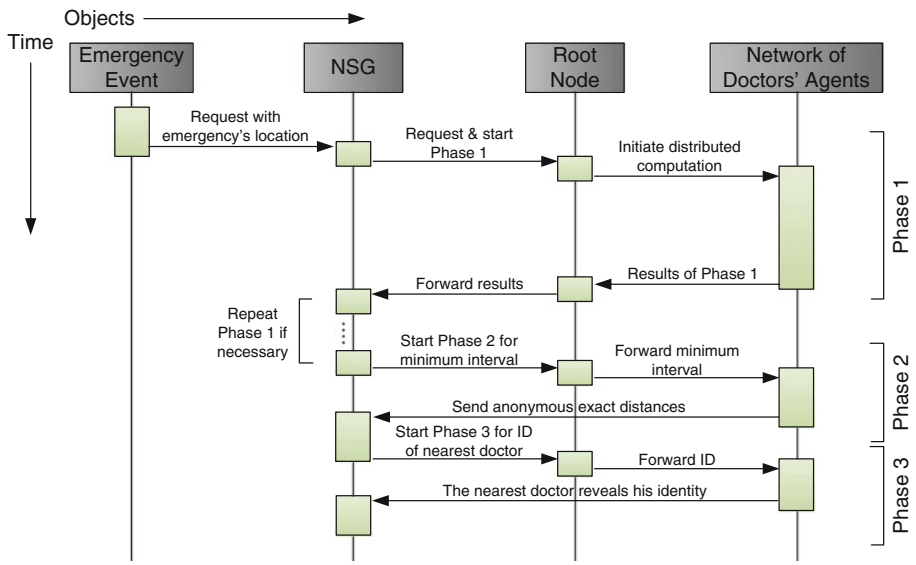


Fig. 2 Interaction diagram of a nearest doctor calculation

- In Phase 1, the closest interval containing at least one doctor is found.
- In Phase 2, the distance of the nearest doctor and an associated random ID are found.
- Finally, in Phase 3, the doctor who owns the random ID realizes that he is the nearest doctor and contacts the NSG to offer his help.

We provide a summary of each phase of the computation.

- **Phase 1**
  - **Input:** The location L_{em} of the emergency.
  - **Output:** An interval I containing the minimum distances in which there is at least one doctor and at most K doctors, where K is a given constant (e.g., $K = 5$).
  - **Description:** The NSG chooses a node as the root node for the particular computation and sends the location L_{em} of the emergency to it. Then, the root node sends a broadcast message to the agent community that starts the distributed protocol and initiates Phase 1. The protocol is executed on a logical binary tree topology that contains all doctor agents (nodes) as leaves (Fig. 3), and some of them also act as intermediate nodes (see Sect. 3.7 for details).

    Phase 1 may last for several rounds. In each round, the root node collects the (intermediate) result of the computation as an encrypted message and sends this message to the NSG. The message is encrypted with the public key of the NSG, which has to be known to all nodes. Let $Count_D$ be the number of distances that belong to the interval of minimum distances. The NSG decrypts the result and obtains

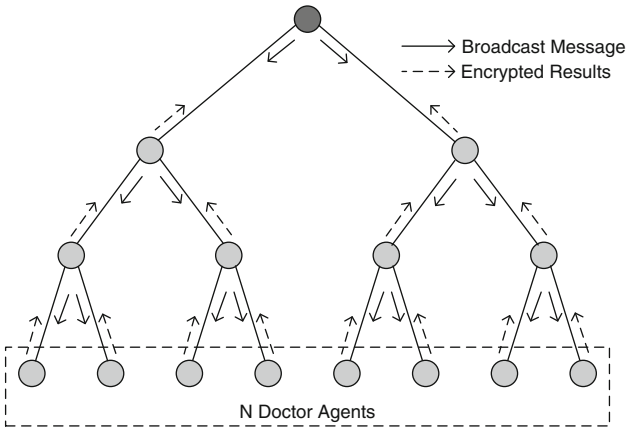


Fig. 3 A binary tree topology

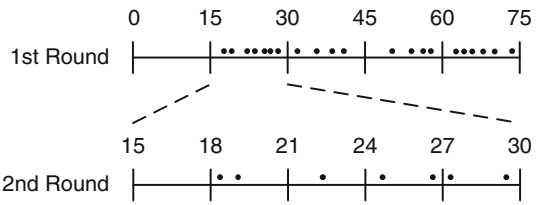


Fig. 4 Example of Phase 1

$Count_D$. If $Count_D > K$, then the computation is repeated in a new round, this time within the interval that has been found. This procedure continues until an interval that contains the closest $Count_D$ doctors, where $1 \leq Count_D < K$, is found. An example of this procedure where the appropriate interval is found in two rounds is shown in Fig. 4. In Sect. 3.5 we describe in detail the protocol of Phase 1 and show that it ensures k-anonymity (see

Definition 5), where $k = N$ and N is the number of all nodes in the network, for the participants of the protocol.

- **Phase 2**
  - **Input:** The interval I from Phase 1.
  - **Output:** The—anonymously collected—exact distances of the $Count_D$ nearest doctors and the associated random IDs.
  - **Description:** In this phase, the NSG sends the interval I of Phase 1 to the root node which in turn sends a broadcast message to announce the interval I to the agents. Each agent whose distance is in the interval I responds by anonymously sending a message to the NSG. The message is encrypted with the public key of the NSG and contains the exact distance of the agent and a random ID (a nonce, i.e., a number used once). For example, by an universally unique identifier (UUID), the probability of collisions is practically negligible. Moreover, even if a collision would occur, the NSG would detect it and repeat this phase. The anonymous transmission is achieved with onion routing [29] techniques. More information about onion routing is given in Sect. 3.6. The NSG collects all anonymous messages and finds the distance of the nearest doctor and the associated random ID. Since the messages are anonymously sent, k-anonymity is preserved in this step; assuming that no background information about the participants of the computation is available, the privacy of doctors is preserved.
- **Phase 3**
  - **Input:** The random ID associated with the distance of the nearest doctor.
  - **Output:** The owner of the random ID realizes that he is the nearest doctor D_n^* and can contact the NSG.
  - **Description:** The NSG sends a message containing the random ID of the distance of the nearest doctor to the root node. The root node broadcasts the ID to the agents' network. The doctor who generated the ID becomes aware of the fact that he is the nearest doctor and contacts directly the NSG.

3.5 A privacy-preserving protocol for Phase 1

We present a cryptographic protocol that finds the first interval of distances in which there is at least one doctor. The protocol uses a trick to encode distance values which has been applied by Yokoo and Suzuki [38] for a secure dynamic programming protocol. Moreover, the cryptographic protocol uses the ElGamal public key cryptosystem [24] and its homomorphic encryption property. Other homomorphic public key cryptosystems like the Paillier cryptosystem [25] can be used in our protocol in place of ElGamal.

Definition 1 (*Homomorphic Encryption*) Homomorphic Encryption is a form of encryption where one can perform a specific algebraic operation on the plaintext[1] by performing an (possibly different) algebraic operation on the ciphertext[2]. The concept was introduced by Rivest et al. [31]. A fully homomorphic cryptosystem [15, 16] supports both addition and multiplication, whereas a partial homomorphic cryptosystem supports only one operation, for example, addition or multiplication.

Definition 2 (*ElGamal Cryptosystem*) The ElGamal Cryptosystem uses an asymmetric key algorithm for public key cryptography which is based on the Diffie-Hellman key agreement[3]. The ElGamal cryptosystem is a partial homomorphic cryptosystem which supports multiplicative homomorphism.

The homomorphic property of the ElGamal cryptosystem is shown in the following equation:

$$\begin{aligned}\mathcal{E}(x_1)\cdot\mathcal{E}(x_2) &= (g^{r_1}, x_1\cdot h^{r_1})(g^{r_2}, x_2\cdot h^{r_2})\\ &= (g^{r_1+r_2}, (x_1\cdot x_2)h^{r_1+r_2}) = \mathcal{E}(x_1\cdot x_2),\end{aligned}$$

where x_1 and x_2 are two plain messages, (G, q, g, h) is the ElGamal public key, G is a cyclic group, q is a large prime integer, g is a generator of the group G, $h = g^x$ is part of the public key and x is the private key, r_1 and r_2 are two random numbers where $r_1, r_2 \in \{0, \ldots, q-1\}$, and $\mathcal{E}(m) = (g^r, m\cdot h^r)$ is the encryption of message m.

3.5.1 The protocol

The protocol accepts three parameters: the minimum distance *minDist*, the maximum distance *maxDist* and the number n of subintervals. These parameters are used to partition the interval of distances (*minDist*, *maxDist*) into a set of n consecutive subintervals. For simplicity, we use subintervals of equal size, but it is straightforward to adapt the approach, for example, to geometrically increasing subintervals. The outcome of the protocol is the first subinterval that contains (the distance of) at least one doctor. In the protocol, each subinterval is represented with a ciphertext, and the whole set of subintervals is represented with the ordered list (or tuple) of the corresponding ciphertexts. Overall, each message has n encrypted numbers,

[1] Plaintext: (in cryptography) an unencrypted message.

[2] Ciphertext: (in cryptography) an encrypted message.

[3] *The Diffie-Hellman key agreement* [10] is the first practical method for establishing a shared secret over an unprotected communications channel.

as many as the subintervals into which the initial interval is partitioned. Such a message containing the ordered list of n ciphertexts passes through each agent.

Each agent prepares its own ordered list of ciphertexts as follows: For doctor D_i, where $i = 1, 2, \ldots, N$, let $\ell_i \in 1, 2, \ldots, n$ be the number of the subinterval that contains the distance of the doctor. Then, the ciphertext for the ℓ_i first subintervals are encryptions of the number "1", and the $\ell_i + 1$ subinterval is encryption of a number z, where $z > 1$ is a fixed value known to all agents. For example, z might be $z = 2$. For the rest of the $n - (\ell_i + 1)$ subintervals, the ciphertexts are encryptions of uniformly chosen random powers of z. An example of a local message is shown in Fig. 5.

When the agent receives the accumulated message, it calculates the new accumulated message as the product of the respective ciphertext of the local message and the accumulated message. The outcome, that is, the new accumulated message, is then forwarded to the next node or nodes.

The distributed computation is performed on a logical binary tree topology in which the leaves of the tree are the N doctors' agents. The depth of the tree is $\lceil \log_2 N \rceil$ and so the accumulated outcome is computed with a reduce operation that requires $\lceil \log_2 N \rceil + 1$ parallel computation steps.

The general form of the final accumulated message is shown in Fig. 6. Let L be the index of the last ciphertext that is an encryption of the number "1". Then, the value of L indicates that the first $L - 1$ subintervals are empty (no doctor is located at a distance within these intervals) and subinterval L is the first non-empty subinterval. The exponent k of the number in the $(L + 1)$-th ciphertext reveals the number of doctors in this subinterval. The

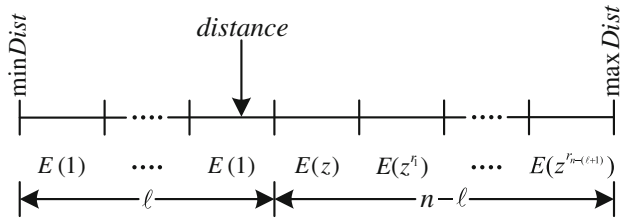


Fig. 5 The local message of a doctor D_i

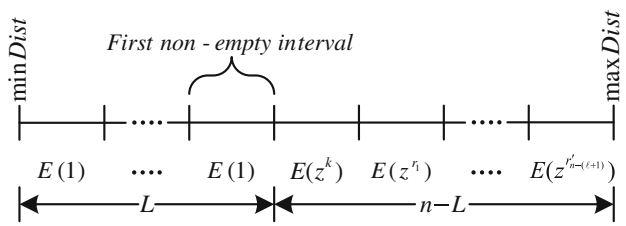


Fig. 6 The final accumulated message

ciphertexts of higher subintervals are encryptions of some random powers of z and are ignored. The NSG decrypts the final message and obtains the first non-empty interval and the number of doctors in it.

3.6 Onion routing

In Phase 2 of the distributed computation, we use onion routing [29], a popular technique for anonymous communication over a network. A simplified description of onion routing is as follows: a node that wants to send a message to another node does not send the message directly to its destination. Instead, the sender chooses a random path that passes through intermediate nodes and terminates at the destination node. Moreover, the sender encrypts the message repeatedly with the keys of the intermediate nodes. So the message is packed with multiple layers of encryption and looks like an "onion". Each intermediate node that receives the message takes away a layer of encryption to reveal routing instructions and sends the message to the next router where this process is repeated. This prevents intermediary nodes from knowing the origin, the destination and the contents of the message.

The advantage of onion routing is that it is not necessary to trust each cooperating onion router (intermediate node); if one or more, but not all, routers are compromised, the anonymity of the communication is preserved. This is because each router in an onion routing network accepts messages, re-encrypts them and then forwards them to another onion router. An attacker with the ability to monitor every onion router in a network might be able to trace the path of a message through the network, but an attacker with more limited capabilities will have difficulty even if he controls one or more onion routers on the message's path.

In order to accomplish the anonymity for sending a message like we described in the Phase 2 (Sect. 3.4), one option is to use the Tor network [11], a widely used, general purpose platform for onion routing. Another option is to implement an onion-like or some other anonymity providing mechanism within the agent community, where each agent forwards its messages through other random agents of the agents community. In the current implementation of the NDP solution, we applied the latter approach (see Sect. 5).

3.7 Network topology

A critical component of the NDP solution is the logical network topology of the agents. The network topology must be scalable, reliable and should support privacy-preserving communications and computations of the agents. We address the above requirements by employing networking

technologies from the field of Peer-to-Peer (P2P) networks. In particular, we apply techniques that have been developed by Stamatelatos et al. [32] and are based on the well-known Chord [33] topology for P2P networks.

The network topology has the following features: The agents are organized into a logical ring that serves as the backbone of the topology. Each node in the ring knows its predecessor and its successor. Actually, for increased tolerance to node changes/failures, each node keeps links to a set of successors. In addition, each node maintains a set of links, called fingers, to nodes at geometrically increasing distances in the ring. These links allow the network to behave as a logical binary tree topology. An example of the embedding of a binary tree topology into the logical ring of the Chord-like architecture is shown in Fig. 7. Similar approaches to embed a virtual tree topology on a chord P2P network are used, for example, by Karger and Ruhl [21].

The proposed network architecture provides a fully decentralized and scalable network topology for the doctors' agents. The links of existing nodes and the establishment of the links of new nodes are accomplished with stabilization procedures that are similar to the typical stabilization procedures of Chord P2P networks.

4 Preserving privacy

In this Section, we examine the security properties of the proposed NDP protocol. We first consider the Honest-But-Curious (HBC) model and show that the protocol preserves the location privacy of the doctors in this model; only a small amount of aggregate or anonymous information is leaked. Then, we examine scenarios with malicious users and discuss how these can be handled.

Definition 3 (*Honest-But-Curious (HBC)*) An Honest-But-Curious party (adversary) [1] follows the prescribed protocol properly, but may keep intermediate computation results, for example, messages exchanged, and try to deduce additional information from them other than the protocol result.

Definition 4 (*Malicious model)* In this model, there are no by default restrictions on what actions an adversary can take; the adversary may behave arbitrarily. It may, for example, submit any value as input to the computation or even abandon the protocol at any step. See the definition given by Kissner and Song [22] or the more detailed treatment by Goldreich et al. [17].

To prove that the protocol preserves privacy, we show that it satisfies the criterion of *k*-anonymity.

Definition 5 (*k-anonymity*) A simple definition of *k*-anonymity [5] in the context of this work is that no less than *k* individual doctors can be associated with a particular personal location.

4.1 Privacy in the HBC model

We first note that the doctors do not use their location in the protocol but only their distance to the location of the emergency. Then, the security of the ElGamal cryptosystem and its homomorphic property ensure that the distances cannot be associated with any particular doctor. Finally, the security of onion routing protects the anonymity of the nearest doctors that disclose their distances in Phase 2. Below, we discuss in detail the preservation of privacy in each phase of the distributed computation.

- **Phase 1**
  - Each doctor uses his private location and the location of the emergency to calculate his distance to the emergency event. Consequently, only the distance is used in the distributed computation, not

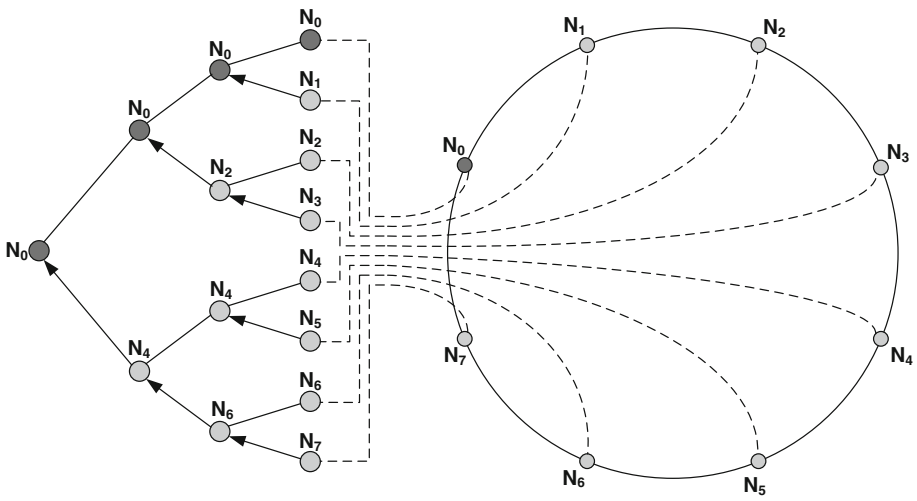


Fig. 7 Embedding of a binary tree into the logical ring

the private location itself. Moreover, the doctor does not use the exact value of his distance but only the subinterval in which this distance belongs.
- Each agent cannot obtain information from the accumulated message that it receives, because the contents of the message are encrypted with the public key of the NSG using ElGamal encryption.
- All ciphertexts of the accumulated message are altered by each node, even the ones that are multiplied with an encryption of the number "1".
- At the end of each round, the final accumulated message reveals the number of doctors in the first interval that contains at least one doctor. No exact location and not even any exact distance is disclosed. Moreover, since no individual doctor can be associated with the doctors in this first interval, Phase 1 preserves k-anonymity, where k is equal to N, the total number of agents in the network. Thus, the aggregate information disclosed in Phase 1 does not violate the location privacy of the doctors.
- Note that each round of Phase 1 is a secure multi-party computation that reveals to the NSG the first non-empty interval and the number of doctors in it. This is an immediate consequence of the security of the ElGamal cryptosystem. Moreover, the first non-empty interval of each round is announced to all doctors (either in Phase 1 or in Phase 2). In conclusion, the outcomes of each round of Phase 1 leak a small amount of aggregate information about the distances of the participating doctors (Table 1).

- **Phase 2**
  - We make the plausible assumption that onion routing works reliably. More details on the security of onion routing can be found in [11]. Then, the security features of onion routing ensure that the exact distances of the doctors in the first interval are anonymously sent to the NSG. Hence, k-anonymity for $k = N$ is preserved in this phase too.
  - We consider the anonymous disclosure of exact distances an acceptable trade-off between efficiency and privacy protection. However, there is the possibility that even an honest NSG may attempt to combine background knowledge with the specific distances to try to identify doctors who live at such a distance from the emergency location. This leakage could be mitigated or avoided if we used less accurate distances in Phase 2 or a more complex protocol among the closest $Count_D$ doctors' agents (see Sect. 4.2.2).
- **Phase 3**
  - In this phase, the random ID associated with the closest distance is announced to the network. The agent that recognizes that it is the owner of this ID can now directly contact the NSG and reveal its identity. The ID does not leak information to any other node.

Table 1 The scope (columns) of the critical data items (rows)

Data items	Participants		
	NSG	Doctors	
		All	$Count_D$-closest
Doctors' location	×	×	×
Emergency location	✔	✔	✔
Phase 1 (each round)			
Closest interval	✔	✔	✔
Number of $Count_D$	✔	×	×
Phase 2			
Exact $Count_D$ distances	✔	×	×
Phase 3			
Nearest doctor D_n^*	✔	×	×

A summary of the critical data items of our solution and to which of the participating entities these items are disclosed is given in Table 1.

4.2 Malicious users

In this section, we examine scenarios with malicious users and discuss how and to what extent the NDP protocol can handle them. The classic results [18, 19] on secure multi-party computation show how to convert a secure multi-party computation of the semi-honest model into a computation in the malicious model. However, the conversion introduces a significant computational overhead since it requires each participant to validate every message by supplying an appropriate zero-knowledge proof that the message is consistent with the protocol specification. In general, a zero-knowledge proof [28] is an interactive method for one party to prove to another that a statement is true, without revealing anything other than the veracity of the statement. This overhead caused by the zero-knowledge proofs raises important practical issues, especially for distributed computations with a large number of nodes as in the case of the NDP.

Instead of using a heavy general conversion technique to handle malicious users, one may consider deriving some proprietary approach for the NDP protocol. In the context of homomorphic encryption, there are examples of

practical problem-specific solutions that can tolerate malicious behavior. For example, in [8, 9] zero-knowledge-based proofs are used within a secure multi-party computation with homomorphic encryption to handle malicious users. However, the above approaches require each node to publish the encryptions of its values to all other nodes of the doctors' network and to execute all the products of ciphertexts. In our case, this is impractical due to the large number of agents.

A straightforward approach could be to handle the complexity of the computationally demanding protocol for malicious nodes by executing the protocol within smaller user groups of NDP nodes (Fig. 8). Such a hybrid solution could be used to assure the tolerance of the NDP protocol against a small number of malicious users. We will describe such an approach in Sect. 4.2.1. Moreover, in Sect. 4.2.2 we present an improvement of Phase 2 to avoid the anonymous disclosure of the smallest exact distances. However, we first examine three specific examples of malicious behavior, and then we consider the hybrid approach as a more general solution to handle malicious users in the NDP protocol.

We shortly discuss the case of a malicious NSG and then proceed to scenarios with malicious nodes. A malicious NSG could collude with any malicious agents to uncover data of neighboring (in a particular computation) agents. Fortunately, such a malicious NSG behavior can be effectively handled by employing a threshold decryption model [6, 14] for the ElGamal Cryptosystem. Using threshold decryption is a classic defense in e-voting systems with the purpose to protect their voting process against malicious coordinators and can be applied within our solution too. In such a case, the NSG instead of merely decrypting the encrypted result uses n parties (the NSG could be one of them) with their secret keys, so that at least t parties, where $t \leq n$, are required to decrypt the final result. We will not further address malicious NSG behavior in this work. Instead, we focus on the community of the doctor agents and examine the following cases of malicious agent behavior.

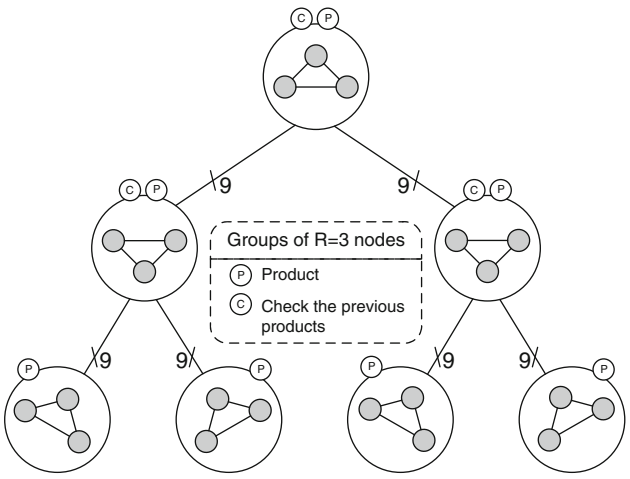


Fig. 8 The execution of Phase 1 on groups of $R = 3$ nodes

- *Case 1:* A user (doctor agent) maliciously or unintentionally reports an erroneous close distance to the emergency incident. As a consequence, this doctor might be wrongly chosen as the nearest doctor, and the actual nearest doctor might not be informed to offer his help.
  A possible solution for this case is to modify the NDP computation of Phase 1 to find a larger number of nearest doctors. For example, the protocol could search for the $Count_D$ nearest doctors, where m is a fixed number and $m \leq Count_D < K$. That is, if the current closest interval contains less than m doctors, then Phase 1 will continue by extending the interval to include larger distances until $Count_D$ is within the specified range. Note that a malicious node would presumably not show up in Phase 3 of the NDP protocol where it has to reveal its identity. Thus, when there are at most $m - 1$ malicious users, the NDP protocol will succeed in finding the actual nearest doctor.
- *Case 2:* A malicious user incorrectly modifies or replaces ciphertexts in the accumulated message of distance intervals of Phase 1. In other words, the user maliciously modifies the accumulated data from the inputs of the preceding users. Note that none of the doctors' agents can see the contents of the accumulated message, but it can change the accumulated message or replace any of its items when the accumulated message is in the possession of the agent. Consequently, the NSG may obtain at the end of the round a wrong number of doctors in the nearest distance interval or even a wrong nearest distance interval.
  The impact of the malicious modification of the ciphertexts depends on many parameters, including the location of the malicious user in the virtual tree of the distributed computation and the difference between the correct and the falsified contents of the ciphertext. The most likely consequence is that the NSG may wrongly decide to proceed or not to Phase 2. In Phase 2 it will become evident that something is wrong if the number of doctors that anonymously reveal their exact distances or the distances themselves does not satisfy the results of Phase 1.
- *Case 3:* A node of the virtual tree topology does not correctly execute the communication operations of the protocol. By definition, a malicious node may not follow the communication steps of the protocol. For example, the node may receive the accumulated

message during the execution of protocol and then not forward this message to next nodes of tree topology. As a result, the distributed computation will not run correctly.

This type of misbehavior can be handled at the network topology level. A fault-tolerant network topology can detect whether the communication is delayed at some nodes and handle these cases as node failures. In the current prototype, we assume that all nodes of the network topology are reliable.

4.2.1 A hybrid approach

In the hybrid approach the agents are organized into groups, where each group consists of R agents. As shown in Fig. 8, each group of agents corresponds to a vertex of the virtual tree topology. The value of R determines the level of tolerance that we wish to achieve against malicious users. We present the hybrid approach in order to give some hints about how to address malicious user behavior in the context of NDP; we do not provide the full details of such a solution.

Within each group of nodes, we can apply a zero-knowledge proof that an encrypted message lies in a given set of messages against malicious behavior. This type of zero-knowledge proofs has been successfully applied in secure e-voting systems [4, 7], and it is straightforward to apply them within our protocol. With such an approach, the R nodes of each group would exchange the encrypted representations of their distance, and each agent would verify that the encrypted values of subintervals lie in the public set $S = \{1, z\}$. No information about the actual value of the encrypted messages would be revealed. We note that we may have to adapt the form of the local encrypted messages in order to simplify the zero-knowledge proofs within the groups. For example, an appropriate form for the local message could be $(1, \ldots, 1, z, 1, \ldots, 1)$. Such an approach will cause higher leakage of aggregate data to the NSG but allows us to simplify the zero-knowledge proof. Since each group of nodes is a clique graph with $R(R-1)/2$ edges and each pair has to exchange a constant number of messages, the overall verification will require $O(R^2)$ communications between the nodes.

Next, each node of the group independently calculates the accumulated message of the group node and sends its output to each of the nodes of the parent vertex in the virtual topology. This step requires R^2 messages. Finally, each of the agents of the parent vertex independently verifies the correctness of the received messages by checking whether the received partial results from the R nodes of the child vertex are identical. Overall, each node of a group will receive messages from $3R-1$ nodes and send messages to $2R-1$ nodes.

If the results that a node receives from the nodes of the child vertex are not identical, this indicates the existence of malicious behavior. The hybrid scheme will successfully detect such malicious behavior if there are at most $R-1$ malicious nodes. Actually, even a much larger number of malicious nodes will be detected as long as at most $R-1$ malicious nodes belong to the same group. Moreover, if at least $R/2+1$ nodes of a child vertex give the same results, then the nodes of the parent vertex may resolve this issue with a simple majority criterion and proceed with the calculations of the NDP protocol.

4.2.2 An improvement for Phase 2

As noted earlier, there is a leakage of data to the NSG in Phase 2, namely the disclosure of the exact distances of the $Count_D$ nearest doctors. Even though the distances are reported anonymously, if the NSG combines them with background knowledge, for example, on the addresses of the doctors, it may potentially identify some of the doctors. We describe a sketch of a solution to the above problem. The solution that we propose is based on a cryptographic protocol—ciphertext comparison [26]—that can compare two ciphertexts without revealing the two encrypted messages. The protocol is valid both in the Honest-But-Curious (HBC) and the Malicious model. The main idea is that a (small) set of independent parties (instead of a single NSG) share the responsibility to coordinate the comparison process and that the correctness of the comparison can be publicly verified.

In the improved Phase 2, the only modification would be that the doctors' agents would have to encrypt their exact distances in an appropriate form given by Peng et al. [26]. This encryption is performed with the common public key of the independent parties. In our case where we have $Count_D$ encrypted distances, in total $Count_D - 1$ secure comparisons are required to find the encryption of the smallest distance or $Count_D \cdot \log(Count_D)$ secure comparisons to sort the encrypted distances. Finally, the random ID of the doctor corresponding to the encrypted smallest distance can then be used in Phase 3. The above comparison process remains efficient for a small number of $Count_D$ distances.

5 Experimental results

To confirm the feasibility of the NDP solution and examine its practical efficiency, we developed and evaluated an NDP prototype. The application is developed in Java, and for the cryptographic primitives, the Bouncycastle [20] library is used. In the prototype, the current location of each doctor is stored in his personal data management

agent, which is an adaptation of the personal agents of the Polis platform developed by Efraimidis et al. [12]. The personal agents use production-ready cryptographic libraries and employ 1024 bits RSA X.509 certificates. The communication between agents is performed over secure sockets (SSL/TLS) with both client and server authentication.

In the next subsections, we first describe a toy-case example with four doctor agents to illustrate the process of execution of the NDP protocol. Then, we present a set of larger-scale experiments with up to 300 doctor agents. In both experiments, we implemented an anonymity mechanism for Phase 2. For simplicity, we used a simple Crowds-like [30] approach where each message is initially passed to a random agent which in turn randomly decides to forward the message either to the receiver or to another randomly chosen agent. More precisely, each doctor agent whose distance belongs to the closest interval identified in Phase 1 prepares a message with its exact distance and encrypts it with the public key of the NSG. The agent then sends this message to another random agent of the agent community. Let us call the recipient, an intermediate agent. Each intermediate agent uses the multiplicative homomorphic property to multiply the encrypted message with the number "1". This changes the ciphertext but leaves the encrypted plaintext unchanged. Then, with probability "1/3" the intermediate agent forwards the message to the NSG and with probability "2/3" to another, random, intermediate agent. Since we assume Honest-But-Curious nodes, this Crowds-like algorithm seems to protect the anonymity of the original sender. However, we do not claim strong security properties of the above algorithm; it simply serves the experimental evaluation of the NDP solution.

5.1 A toy-case experiment

In this experiment, the NSG submits an emergency event to a community of four doctor agents. The NDP protocol is then used to identify the doctor who is the closest to the emergency. The doctor agents are assumed to be Honest-But-Curious (HBC), and there are no network or agent failures. The agent community is organized into a simple ring topology.

The experiment covers a hypothetic square area of 10,000 km^2. The locations of the doctors and the emergency are chosen independently and uniformly at random in the above area. Each agent chooses its random location, and the NSG chooses a random location for the emergency event. The NDP solution tries to find the nearest doctor within a distance of at most 75 km from the emergency location. The values of the internal parameters of the NDP solution are $K = 2$ and $z = 2$, where K is the upper bound of the closest doctors of Phase 1 and z is a fixed number used in the encryptions of the intervals.

The NSG chooses an agent node, in this case "*Agent*_1", as the root node and forwards the location of the emergency event to this node. The coordinates of the location of the emergency of the experiment are $L_{\text{em}} = [41.140110, 24.913660]$, and the exact distances of the four agents from this emergency location are the following:

$Agent\_1 \Rightarrow 17.544817$ km
$Agent\_2 \Rightarrow 53.157742$ km
$Agent\_3 \Rightarrow 25.797003$ km
$Agent\_4 \Rightarrow 66.221868$ km

The cryptographic protocol starts with Phase 1. In the first round the interval of distances (in km) [0, 75] is partitioned into five equal intervals. As a result, the encrypted representation of agents' distance (in km) is as follows:

	0–15	15–30	30–45	45–60	60–75
Agent_1	E(1)	E(1)	E(2)	E(2^5)	E(2^3)
Agent_2	E(1)	E(1)	E(1)	E(1)	E(2)
Agent_3	E(1)	E(1)	E(2)	E(2^6)	E(2^9)
Agent_4	E(1)	E(1)	E(1)	E(1)	E(1)

The final accumulated message in round 1 is shown below:

	0–15	15–30	30–45	45–60	60–75
Result	E(1)	E(1)	E(2^2)	E(2^{11})	E(2^{13})

The decryption of the ciphertexts of the final message reveals that the first non-empty interval is [15,30) and that this interval contains two doctors. Since the number of doctors in the first interval is equal or less than K, Phase 1 terminates. In Phase 2, the root node broadcasts the first interval to all nodes. Every node with a distance in this interval anonymously sends its exact distance together with a random ID nonce (number used once) to the NSG.

The NSG receives the following two exact distances and the associated random ID numbers:

$[\text{Dist} = 17.544817, \text{ID} = 56770656]$
$[\text{Dist} = 25.797003, \text{ID} = 45413392]$

The NSG finds that the minimum distance is 17.544817 km. In Phase 3, the NSG sends the random ID nonce that is associated with the minimum distance to the

root node, which in turn broadcasts this ID to the doctors' network.

ID : 56770656

Finally, "*Agent*_1" realizes that it is the nearest doctor and directly contacts the NSG to offer its services. A snapshot of the application GUI during the experiment is shown in Fig. 9.

5.2 The large-scale experiment

We also conducted a set of experiments with a gradually increasing number of agents from 50 to 300 agents. Also in this case, we assume that the doctor agents are Honest-But-Curious (HBC), and there are no network or agent failures. The experiments were executed on a 100-Mbps network with 30 workstations, each with a CPU Intel Core 2 Quad Q8300 processor at 2.5 GHz and 2 GB RAM. The agents were distributed evenly on the available workstations so that each workstation ran at most 10 agents. Every measurement was averaged on 10 independent executions of the experiment, each with different random values for the location of emergency and the locations of doctors.

The network topology is a logical binary tree. The root of the tree is the root node of the particular NDP computation. At this stage of the prototype development, the tree topology is build by the NSG or by a Directory Service (DS) where every active agent registers itself. Consequently, we do not rely on the Chord-like network to build the tree topology as a full-blown implementation of the NDP solution would do.

The measurements concern the execution time of each phase of the NDP protocol and the total run time. We present the results and derive conclusions about the efficiency and the scalability of the solution. In Fig. 10, the execution times of a single run of Phase 1 are shown. Recall that Phase 1 may be executed more than once, depending on how many doctors are found in the closest non-empty interval. In this figure we focus on the time for a single round of Phase 1. The total run time for all repetitions of Phase 1 in an NDP computation is taken into account in Fig. 12 that presents running times of the complete NDP computation.

From Fig. 10 we conclude that the execution time of Phase 1 depends almost linearly on the depth of tree topology of the distributed computation. For example, for an NDP computation with 100 agents, the depth of the underlying tree topology is $\lceil \log_2 100 \rceil = 7$, whereas for computations with 150, 200 or 250 agents, the depth is $\lceil \log_2 150 \rceil = \lceil \log_2 200 \rceil = \lceil \log_2 250 \rceil = 8$. The results show that Phase 1 scales well as the number of agents increases.

Figure 11a, b shows the execution times of Phases 2 and 3, respectively. For Phase 2, the results show that its execution time does not seem to depend on the number of agents. This is rather expected, since the workload of Phase 2 mainly depends on the number of nearest doctors accrued in Phase 1 and how fast these doctors can anonymously send their exact distances to the NSG. The results of Phase 3 show that the execution time of this phase is dominated by a broadcast operation, which in turn depends on the depth of the virtual tree topology.

Finally, in Fig. 12 the overall results are presented. In particular, the execution time of Phase 1 (of a single execution of the phase), Phase 2, Phase 3, the sum of the previous times and the total time of the whole NDP computation are presented. Note that the total time may differ from the sum of the three phases. This happens in cases where Phase 1 has to be executed more than once within the same computation.

The general conclusions of this large-scale experiment are that the NDP protocol can be successfully performed on a virtual tree topology and that, as expected, the execution times seem to increase linearly with the depth of tree

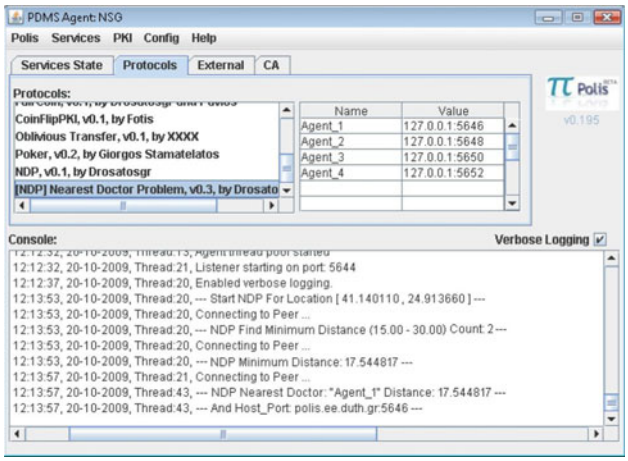


Fig. 9 A snapshot of the NSG (NDP Service Gateway)

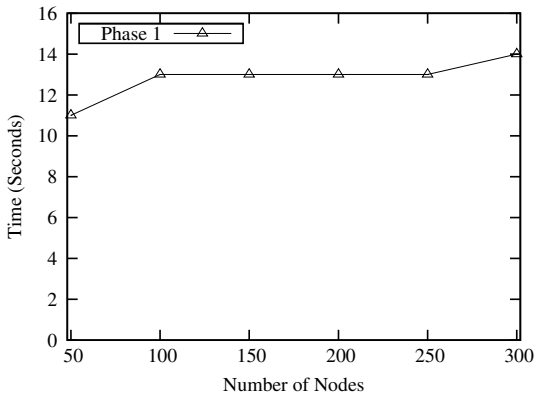
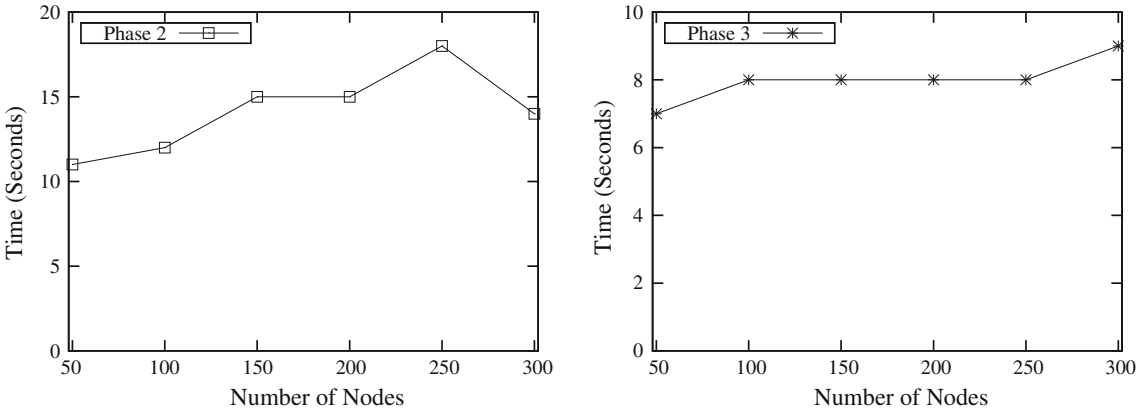


Fig. 10 Execution times of Phase 1 with respect to the number of agents



(a) Execution times of Phase 2. **(b)** Execution times of Phase 3.

Fig. 11 Execution times of Phase 2 and Phase 3 with respect to the number of agents

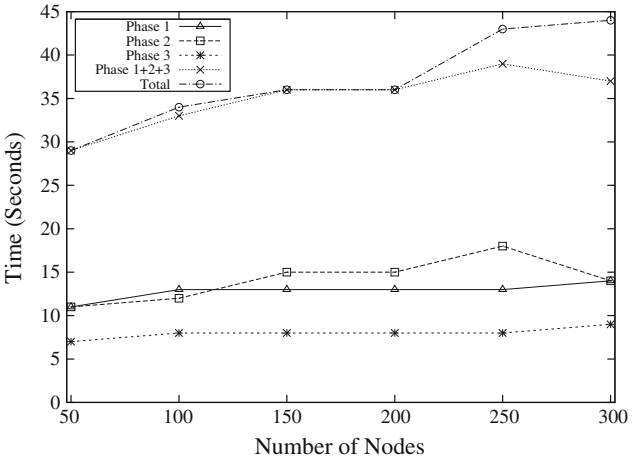


Fig. 12 Execution times of Phase 1, Phase 2, Phase 3, sum of 3 phases and total time with respect to the number of agents

topology and thus logarithmically with the number of nodes. Based on the design of the NDP protocol and the above experimental results, we are confident that the NDP solution can scale well to handle much larger communities with thousands of nodes.

6 Discussion

The development, analysis and evaluation of the NDP prototype confirmed the feasibility and the effectiveness of the NDP solution. However, there are still important open questions, technical and non-technical. A critical technical issue is the robustness of the network topology against failures. In a community with a large number of nodes, the short-term or long-term failure of individual nodes or network links will be a common event, and the P2P-based network topology should be able to handle these failures. A lookup of the nearest doctor may take too long if the size of the doctors' community is large and the network is currently recovering from some failure. Furthermore, a temporary node or link failure, which becomes more likely as the network size increases, can disrupt the whole distributed computation. We believe that a fully developed network platform based on Chord-like peer-to-peer networking techniques can address these technical issues of the network topology.

At the service level, the NDP solution could be improved with a more appropriate distance metric. In the prototype, we used the great-circle distance. However, for inhabited areas, a much better distance metric could be the time that each doctor needs to reach to the emergency location. For example, a navigation software tool running at the agents' side could use the GPS location, updated maps and possibly the current traffic conditions to calculate an estimate of the time that the doctor will need to arrive at the location of the emergency. Such a distance metric would be much more effective for the NDP.

Another important point is that even though wireless communication options like 3G, Wi-Fi and Satellite communications are now widely available, there are still technical and economic issues. For instance, a personal mobile device will have to regularly update the doctor's location at his personal data management agent. This may cause the energy consumption of the mobile device and the cost for the wireless data transfer to become prohibitive. However, the current momentum of mobile device technology and telecommunications services predispose that these issues will soon be overcome.

Finally, we would like to emphasize an issue, which may not be technical, but is of equal importance for the acceptance of a solution like NDP by real doctor communities or other communities that could use such a system. Clearly, many individuals may not be eager to adopt a

technology like the NDP solution in their everyday life. However, such difficulties commonly exist for every new technology. We believe that the doctors' community should not feel threatened in any way by an application like the NDP solution, because:

1. The privacy of each doctor's location is preserved and under the absolute control of the doctor.
2. The solution is simple and cheap enough to be feasible even with current information and communication technologies (ICT).
3. The benefits of an application like NDP for the public well-being are practically immeasurable.

7 Conclusion

In this paper, we proposed the use of the current locations of participating doctors for supporting services for the public well-being. For this reason, we define the Nearest Doctor Problem (NDP) and make a first attempt to present an efficient privacy-preserving protocol for solving it. The proposed scheme utilizes the doctors' personal data (location) while ensuring their privacy. The protection of privacy is achieved by using cryptographic techniques and performing a distributed computation within a network of personal agents. Furthermore, we studied how our proposed approach can be applied under malicious users and suggested possible countermeasures. For the feasibility of our approach, we developed a prototype implementation and confirmed the viability and the efficiency of the proposed solution by conducting a set of experiments with up to several hundred doctor agents.

In our view, the NDP solution for offering help in case of an emergency should be considered a complement to existing emergency handling services. The NDP solution would probably make a difference only in some cases of emergencies. However, even a small number of successful applications of NDP justify, at least in our view, the approach.

A future direction for the improvement of our solution could be to give a more precise security and privacy analysis of our protocol. Even though only a small amount of aggregate or anonymous information is leaked by the intermediate results of the computation, a formal estimation of this leakage would be an important step for this work. Even more interesting would be to obtain a solution with no side leakage at all, essentially a secure multi-party computation for the NDP. Finally, an interesting extension of the NDP would be to require the location of the emergency to be private, too.

References

1. Acquisti A, Gritzalis S, Lambrinoudakis C, De Capitani di Vimercati S (2008) Digital privacy. Auerbach Publications, Taylor & Francis Group, 6000 Broken Sound ParkWay NW
2. Bickson D, Dolev D, Bezman G, Pinkas B (2008) Peer-to-peer secure multi-party numerical computation. IEEE international conference on Peer-to-Peer computing, pp 257–266
3. Bogetoft P, Christensen DL, Damgård I, Geisler M, Jakobsen T, Krøigaard M, Nielsen JD, Nielsen JB, Nielsen K, Pagter J, Schwartzbach M, Toft T (2009) Secure multiparty computation goes live. In: Dingledine R, Golle P (eds) Financial cryptography and data security. Springer, Berlin, pp 325–343
4. Brandt F (2005) Efficient cryptographic protocol design based on distributed el gamal encryption. In: Proceedings of the 8th international conference on information security and cryptology (ICISC 2005), vol 3935. Springer, LNCS, pp 32–47
5. Ciriani V, Capitani di Vimercati S, Foresti S, Samarati P (2007) κ-anonymity. In: Yu T, Jajodia S (eds) Secure data management in decentralized systems, advances in information security, vol 33. Springer, Berlin, pp 323–353
6. Clarkson M, Chong S, Myers A (2008) Civitas: Toward a secure voting system. In: IEEE symposium on security and privacy (SP 2008), pp 354–368
7. Cramer R, Gennaro R, Schoenmakers B (1997) A secure and optimally efficient multi-authority election scheme. In: Proceedings of the 16th annual international conference on theory and application of cryptographic techniques. Springer, Berlin, EUROCRYPT'97, pp 103–118
8. Cramer R, Damgård I, Nielsen JB (2001) Multiparty computation from threshold homomorphic encryption. In: Proceedings of the international conference on the theory and application of cryptographic techniques: advances in cryptology. Springer, London, EUROCRYPT'01, pp 280–299
9. Damgård I, Nielsen J (2003) Universally composable efficient multiparty computation from threshold homomorphic encryption. In: Advances in cryptology—CRYPTO'03 (Lecture notes in computer science), vol 2729. Springer, Berlin, pp 247–264
10. Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–654
11. Dingledine R, Mathewson N, Syverson P (2004) Tor: the second-generation onion router. In: Proceedings of the 13th USENIX security symposium, pp 303–320
12. Efraimidis PS, Drosatos G, Nalbadis F, Tasidou A (2009) Towards privacy in personal data management. J Inf Manag Comput Secur 17(4):311–329
13. Europe's Information Society (2009) eSafety. URL http://ec.europa.eu/esafety
14. Gennaro R, Jarecki S, Krawczyk H, Rabin T (2007) Secure distributed key generation for discrete-log based cryptosystems. J Cryptol 20:51–83
15. Gentry C (2009) Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on theory of computing, ACM, New York, NY, USA, STOC'09, pp 169–178
16. Gentry C (2010) Computing arbitrary functions of encrypted data. Commun ACM 53:97–105
17. Goldreich O (2004) The foundations of cryptography, vol 2. Cambridge University Press, Cambridge
18. Goldreich O, Micali S, Wigderson A (1987) How to play any mental game or a completeness theorem for protocols with honest majority. In: Proceedings of the nineteenth annual ACM symposium on Theory of computing, ACM, New York, NY, USA, STOC'87, pp 218–229

19. Goldreich O, Micali S, Wigderson A (1987) How to prove all NP statements in zero-knowledge and a methodology of cryptographic protocol design (extended abstract). In: Odlyzko A (ed) Advances in cryptology—CRYPTO'86 (Lecture notes in computer science), vol 263. Springer, Berlin, pp 171–185
20. Hook D (2005) Beginning cryptography with Java. Wiley Publishing Inc., Indianapolis
21. Karger DR, Ruhl M (2004) Diminished chord: a protocol for heterogeneous subgroup formation in peer-to-peer networks. In: Proceedings of the third international conference on peer-to-peer systems. Springer-Verlag, Berlin, Heidelberg, IPTPS'04, pp 288–297
22. Kissner L, Song D (2005) Privacy-preserving set operations. In: Shoup V (ed) Advances in cryptology—CRYPTO'05 (Lecture notes in computer science), vol 3621. Springer, Berlin, pp 241–257
23. Lindell Y, Pinkas B (2009) Secure multiparty computation for privacy-preserving data mining. Confidentiality 1:59–98
24. Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Inc., Boca Raton
25. Paillier P (1999) Public-key cryptosystems based on composite degree residue classes. In: Stern J (ed) Advances in cryptology—EUROCRYPT 1999, vol 1592. Springer, Berlin, Heidelberg, LNCS, pp 223–238
26. Peng K, Boyd C, Dawson E, Lee B (2005) Ciphertext comparison, a new solution to the millionaire problem. In: Proceedings of the 7th international conference on information and communications security (ICICS 2005), vol 3783. Springer, LNCS, pp 84–96
27. Priyantha NB, Chakraborty A, Balakrishnan H (2000) The Cricket location-support system. In: MobiCom '00: Proceedings of the 6th annual international conference on mobile computing and networking, ACM, New York, NY, USA, pp 32–43
28. Quisquater JJ, Guillou L, Annick M, Berson T (1989) How to explain zero-knowledge protocols to your children. In: Proceedings on advances in cryptology, Springer New York, Inc., New York, NY, USA, CRYPTO'89, pp 628–631
29. Reed M, Syverson P, Goldschlag D (1998) Anonymous connections and onion routing. IEEE J Sel Areas Commun 16(4):482–494
30. Reiter MK, Rubin AD (1998) Crowds: anonymity for web transactions. ACM Trans Inf Syst Secur 1:66–92
31. Rivest R, Adleman L, Dertouzos M (1978) On data banks and privacy homomorphisms. In: DeMillo R, Dobkin D, Jones A, Lipton R (eds) Foundations of secure computation. Academic Press, New York, pp 169–177
32. Stamatelatos G, Drosatos G, Efraimidis PS (2009) Quantum: A peer-to-peer network for distributed computations with enhanced privacy. In: EYRHKA 2009 conference proceedings, 3rd Panhellenic scientific student conference on informatics, pp 201–210, written in Modern Greek
33. Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H (2001) Chord: A scalable peer-to-peer lookup service for internet applications. In: ACM SIGCOMM'01, San Diego, CA, pp 149–160
34. To Vima Online (2007) Missing doctor incident. URL http://www.tovima.gr/relatedarticles/article/?aid=209954
35. Want R, Hopper A, Falcão V, Gibbons J (1992) The active badge location system. ACM Trans Inf Syst (TOIS) 10:91–102
36. Ward A, Jones A, Hopper A (1997) A new location technique for the active office. IEEE Pers Commun 4(5):42–47
37. Yao ACC (1982) Protocols for secure computations (extended abstract). In: Proceedings of twenty-third IEEE symposium on foundations of computer science, Chicago, Illinois, pp 160–164
38. Yokoo M, Suzuki K (2002) Secure multi-agent dynamic programming based on homomorphic encryption and its application to combinatorial auctions. In: Proceedings of the first international joint conference on Autonomous agents and multi-agent systems: part 1, ACM, New York, NY, USA, AAMAS'02, pp 112–119