Secure Bootstrapping of Distributed Hash Tables in Dynamic Wireless Networks

L. Cheng, K. Jean, R. Ocampo, A. Galis
Department of Electronic and Electrical Engineering,
University College London,
London, UK, WC1E 7JE.

P. Kersch, R. Szabo
Department of Telecoms and Media,
Budapest University of Technology and Economics,
Budapest, Hungary.

Abstract—Existing research work on Distributed Hash Tables (DHTs) assume that, prior to actually establishing a DHT, large number potential member nodes would come together at one point and at one place, and would all agree on the characteristics of the DHT to-be-established. This paper presents DHT bootstrapping, which is a novel approach to enable secure, efficient and scalable deployment of DHTs in wireless networks, in particular in wireless networks with time-varying topology. Our solution is not restricted to a particular type of DHT implementation.

Keywords—Bootstrapping; Distributed Hash Tables (DHTs); DHT bridging; wireless ad-hoc networks; zone division.

I. INTRODUCTION

Distributed Hash Tables (DHTs) enable distributed and scalable data storage and lookup services. They also support distributed management systems in large-scale networks. Existing DHT implementations such as Content Addressable Networks (CANs) [1], Pastry [2], Chord [3], Tapestry [6] and more, were originally designed for the Internet. In recent years, a significant number of proposals have been made to adapt existing DHT implementations, so that they can be deployed in wireless ad-hoc environments with time-varying network topology, i.e. Mobile Ad-hoc NETWORKs (MANETs), Ambient Networks (ANs) [14], or other dynamically changing networks. For example, a protocol for establishing Pastry DHTs in MANETs was proposed in [4]; the protocol aims at optimising routing locality in the resultant DHT by utilising network-layer routing information. In [5], an integrated approach was presented to integrate Pastry with a multi-hop routing protocol, to enable more efficient DHT establishment in MANETs, etc. Existing research works usually have one assumption: that physical connectivity have been established between a large number of participating wireless ad-hoc nodes; and through this physical connectivity, these nodes somehow have all expressed their willingness to become members of a DHT of certain agreed characteristics. However, wireless ad-hoc nodes have their own preferences on the characteristics of the to-be-established DHT; also wireless ad-hoc nodes may move around. Little work has been conducted how nodes can come up with an agreement, and subsequently setup a DHT (based on the agreement) in dynamically changing and heterogeneous (medium to large-scale) mobile ad-hoc networks.

This paper suggests that a DHT deployment protocol for mobile ad-hoc networks must take into account the dynamicity, heterogeneity, and scalability factors. In addition, security should be provisioned. In this paper, a novel solution to the problem space, known as DHT bootstrapping, is presented. DHT bootstrapping investigates the enabling of DHT deployment in wireless ad-hoc networks from a different angle: prior to actually using the standard or refined DHT protocols to establish DHTs in wireless ad-hoc networks, there is a need of a mechanism to arrange potential member nodes (of the to-be-established DHT) in a structured manner to enable the resolution of heterogeneity, dynamicity and scalability issues in the network. Security is provisioned in the protocol.

II. BACKGROUND

To realise the need of DHT bootstrapping in dynamic wireless networks, here, a deployment scenario of DHT in dynamic wireless networks is described. Two very pertinent applications of DHTs in dynamic wireless networks are in the location of distributed context in wireless networks and in peer-to-peer Session Initiation Protocol (SIP) [15]. Distributed context [17] refers to the situational information about entities in a distributed network. SIP [16] is an application layer protocol concerned with the creation, management and termination of multimedia sessions across networks. In its distributed form (peer-to-peer SIP), the protocol operates without the need for any centralised servers for the location of SIP endpoints and the routing of session management messages. This location and routing can be performed using DHTs. Furthermore, the use of the distributed context information can be used to enhance the SIP services resulting in personalisation according to network state, end user device or end user situation. A DHT can be used to store location mappings for the context information stored in the network thus allowing its effortless location, acquiring and use [13]. As a result, in dynamic wireless networks, the bootstrapping of DHTs can prove valuable both in the provision of peer to peer SIP services and for the location of distributed context. These

1 Some nodes may be installed with different types or numbers of DHT software, i.e. some may have multiple DHT software installed and thus support multiple types of DHT implementations (e.g. CAN, Pastry, Chord...), but some may support one type of DHT implementation only.
two applications nicely complement each other as context can be used to enhance SIP services whilst context can be carried in SIP message bodies.

However, to enable such (distributed) context-aware facility in a large-scale, heterogeneous mobile ad-hoc network, that are either directly or indirectly physically connected (we will discuss more about sizes of bootstrapping networks later on); there is a need to enable distributed data storage (through the use of a DHT). Hence, DHT bootstrapping must be addressed. The nodes forming this distributed data storage facility are known as the participating nodes in this paper, because they shall participate in the (subsequent) DHT establishment process. Prior to the actual DHT establishment, participating nodes must come to an agreement on the characteristics of the to-be-established DHT, known as a DHT agreement in this paper (e.g. keysize, hash algorithms, etc.). This negotiation is possibly conducted through multicast/broadcast. Pre-defining a DHT agreement among all potentially participating wireless ad-hoc nodes restricts the level of flexibility to serve individual’s specific needs (i.e. some node’s with specific requirements may be ignored), and is neither practical nor scalable. On the other hand, node dynamicity has a significant impact on the applicability and efficiency of real-time negotiations. Node dynamicity means, potentially, the negotiation process might result in a loop. Once a confirmed DHT agreement has been reached between all participating nodes, the next step would be to decide which one of the participating nodes to instantiate the DHT establishment process i.e. to become an instantiation node. However, to determine an instantiation node, it means that in additional to the DHT implementation negotiation process mentioned earlier, either an additional round of real-time peer election/selection is needed, or the election/selection process must be conducted together with the DHT implementation negotiation process. In either case, additional overhead would be incurred. Thus, there is a need to investigate a technique to minimise this additional overhead. In addition, there is a need to investigate how to enable more efficient and scalable, preferably more distributed DHT joining processes. Furthermore, assuming a DHT is established per (medium to large) network, ideally, when these networks come to close physical proximity, their DHTs should be merged together to enable data sharing. Therefore, there is a need to investigate an efficient technique to enable data sharing between DHTs of different composing wireless ad-hoc networks. Lastly, security should be provisioned.

III. DHT Bootstrapping

In this section, the DHT bootstrapping protocol set will be presented. The entire process has three stages, and it is composed of three protocols: the DHT negotiation protocol, the DHT establishment protocol, and the DHT bridging protocol. For simplicity, we assume all nodes in a wireless ad-hoc network are willing to become members of the to-be-established DHT, as long as their preferences on DHT implementations are satisfied (i.e. all nodes in a network are participating nodes). We assume that nodes know at least their immediate physical neighbours, and can route packet to other nodes. We assume all links are ad-hoc links (ad-hoc links have limited physical range). We assume node mobility is low to medium. Extremely high node mobility is possible in a real network, but if a node is moving across networks rapidly, it is not necessary to create association between a fast moving node with other nodes. Node mobility will be discussed later on.

A. The DHT Negotiation Protocol

A large network is divided into zones. Each zone is a virtual network. For simplicity, we assume zones do not overlap with each other (that each participant node is a member of one zone only). Zones are divided based on the network’s physical topology, of which nodes within a zone (i.e. the zone nodes) are physically near to each other. The size of each zone (i.e. the number of nodes in each zone) is variable (see later). Each zone has at least one automatically elected zone head. A zone head is a node that is responsible for certain specific management operations related to DHT establishment within the zone. The zone head itself, however, does not have to be a candidate member of the to-be-established DHT (see later). Zone heads in the entire network together form a virtual backbone network that supports cross-zone DHT operations (Figure 1a). We will discuss more on zone division, zone head election, and zone heads’ operations later on. An example of zones (zone a to d) and the virtual backbone is shown in Figure 1a.

Zone nodes within each zone carry out DHT implementation negotiation, say, by multicasting within the zone; alternatively, the zone head(s) may query members of the zone respectively to find out their preference(s) on the resultant DHT. Thus, instead of having one DHT negotiation across the entire network, several DHT implementation negotiations are carried out across the entire network simultaneously. Note that each negotiation is conducted within a zone, which means the number of nodes participating in the negotiation is less (than the total number of nodes in the network). With fewer participants, negotiation is speeded up to minimise the chances of new nodes joining or participating nodes dropping out. We will discuss how our solution handles node dynamicity later on. In a heterogeneous environment, the majority of zone nodes within a zone will come to a DHT agreement; the rest, however, would be excluded due to incompatibility. Those nodes that can come to an agreement (i.e. the member nodes) will form a DHT between themselves based on the agreed DHT characteristics (see later for DHT establishment). Note that DHT is established per zone. On the other hand, those that cannot come to an agreement (i.e. the pending nodes) will not be included in this initial stage of DHT establishment. The
pending nodes will be served at a later stage (see later). Note further that once the negotiation process has begun, new joining nodes will not be allowed to take part in the negotiation process. However, similar to the pending nodes (i.e. those nodes that took part in the negotiation process but cannot come to an agreement with other nodes in the same zone), they will be handled at a later stage (see later). Figure 1b shows different nodes in a network.

B. The DHT Establishment Protocol

Within a zone, the member nodes will create a DHT among themselves. The DHT establishment process will be instantiated by (one of) the zone head(s). If there are more than one zone head among the member nodes (i.e. zone a in Figure 1b), the zone heads will determine among themselves which to instantiate the DHT establishment process. If none of the member nodes is a zone head (i.e. zone b in Figure 1b), the zone head appoints a member node that is physically nearest to itself to instantiate the DHT establishment process. The appointed member node would become a zone head, and it is also the instantiation node. Routing locality is provisioned in our solution. Note that because a DHT is now established within a zone, of which nodes are physically near to each other, routing locality is therefore partially achieved in the sense that messages can be routed within a zone i.e. within a DHT of which nodes are physically nearest to each other (reduced routing delay). There are several reasonable DHT establishment approaches, which can be adopted to enhance further routing locality. For example, the MADPastry approach [4] uses random landmarking [7] to create DHTs for nodes that are physically near to each other. We have developed the absorption approach [8] that achieves routing locality by enabling new joining nodes to own keyspaces that are immediate neighbours (on the DHT overlay) to keyspaces owned by their immediate physical neighbours. Thus, the DHT overlay routing topology would be more coherent with the underlying physical routing topology. We will discuss more on how DHT bootstrapping provisions for routing locality later on.

C. The DHT Bridging Protocol

Note that the result of DHT bootstrapping would be several to many DHTs being established within one wireless network (i.e. one DHT per zone). To enable distributed data storage and lookup, there is actually no need to establish one DHT per network, as long as these established DHTs allow clients to search all their stored data in an efficient manner i.e. cross-DHT searching. The next stage of DHT bootstrapping is therefore to create bridging between these DHTs, so that a data search request originated from a client of one zone will be forwarded simultaneously to other zones, where the search will also be conducted. This is known as DHT bridging and simultaneous searching. DHT bridging means that the zone head(s) of one zone/DHT are made aware of the zone heads of its (theirs) immediate physical neighbouring zones/DHTs (see later). When a search is originated by a client of a zone, the search will be conducted within the originating zone (using the standard DHT searching algorithm); but simultaneously, the search request is forwarded through the originating zone’s zone head (if multiple zone heads exist, the client will forward its request to the zone head that is nearest to itself), and subsequently forwarded to the zone heads of the originating zone’s immediate neighbouring zones, where the search will be conducted simultaneously, and so on (hence the name simultaneous search). State maintenance on zone heads is minimised by requiring zone heads to keep information of the zone heads of their immediate neighbouring zones only. The virtual backbone network formed by zone heads of individual zones is where searches will be forwarded between zones. To minimise traffic through the backbone, successful search results are sent directly back to the client, instead of routing through the backbone (we assumed all nodes are either directly or indirectly physically connected on one network). Note that simultaneous search does not necessary mean multiple search results will be returned. The number of search results returned depends on the availability of the searched item in the network.

Note that the bridged DHTs are assorted. By assorted DHTs, we mean the DHTs may use different implementations (i.e. CAN-DHTs, Chord-DHTs, Pastry-DHTs, and more), and may use different keysize (i.e. 160-bit, 256-bit, and more). To enable simultaneous search across assorted DHTs, the original search request must be mapped into the correct format. For example, if the original search request was originated from zone 1 that uses a 160-bit CAN-DHT, to conduct the same search in a 256-bit Chord-DHT (e.g. zone 2), the request must be hashed using the correct hash algorithm and keysize (e.g. SHA-256). This mapping is conducted at the zone head of the to-be-searched DHT. The mapping zone head can distribute mapping load to its member node is desired. We will discuss how DHT bridging is also applicable to sharing data between composing networks/DHTs later on.

D. Pending Nodes & New Joining Nodes Management

We have mentioned that some nodes in a zone are not allowed to take part in the DHT establishment process i.e. the pending nodes and the new joining nodes. Instead of excluding these nodes, in DHT bootstrapping, these nodes register themselves with a member node that is physically nearest to them. This registration includes basic information such as location identifiers (IP in this case), and the nodes’ preferences on the characteristics of the DHT that they wish to join with. Once registered, these nodes become the registered nodes. A registered node is one that has registered itself with a member node (or with another registered node, see shortly later), but itself is yet to become a member node (note that when it eventually becomes a member node, it does not necessary to be a member of the DHT of the zone that it has registered with, see shortly later). A registered node is allowed to accept registrations from other pending or new joining nodes. As such, we create a completely distributed registration process: we do
not have a single point of registration, but every node (as long as they are member nodes or registered nodes) is capable of accepting registrations. As more new nodes are registered, more nodes would become capable of taking registrations from other pending or new joining nodes. Figure 2a shows the pending nodes, new joining nodes, and member nodes in different zones in a network. Figure 2b shows how distributed registration is carried out in a network.

Once the DHTs have been established in the network, member nodes that hold registrations will determine whether the registered requirements meet the characteristics of the local DHT (i.e. the DHT of which the member nodes are members of). If they match, the corresponding registered node will be invited to join the local DHT. If they do not match, the registrations will be forwarded as joining requests to other zones through the virtual backbone network. The same applies to all nodes that hold registrations. The joining requests will be intercepted at (remote) zone heads of appropriate zones (i.e. zones that host DHTs that fulfill the specific DHT implementation requirements specified in the joining requests), and the joining process will be conducted between the corresponding registered node and an existing member of the DHT. As a result, the registered nodes will eventually become members of a DHT. However, unlike the others, these nodes do not take part in the initial negotiation process. As such, they may join a DHT that is remote from their physical locations. One may argue that this arrangement would degrade a certain level of routing locality. It is not within the scope of this paper to develop a specially designed solution to optimise routing locality in DHTs; but our solution provisions for routing locality. By requiring DHTs to be established per zone, the majority of DHT members are physically near to each other, hence routing locality is partially optimised (see later for more on routing locality). In a rare situation where some joining requests cannot be accommodated by any existing DHTs (say, the DHT implementation requirements are less popular), the registered node would be notified of join failure (by the node that holds its registration), or wait for a timeout, and start instantiating its own DHT with its own specific requirements if it feels that there is still a need to have such specific DHT. Through this mechanism that handles pending nodes and new joining nodes, potential indefinite looping in DHT negotiation is avoided.

IV. Protocol Design

A. Scalable and Efficient DHT Negotiation, DHT Establishment, and DHT Composition

We have mentioned that a network is divided into zones based on the physical network topology. The idea of establishing one DHT per zone, rather than establishing one DHT over the entire network, is to reduce the time delay in the overall DHT establishment process. We have also discussed that in a wireless ad-hoc network of variable shape, nodes that locate physically far away from the instantiation node might have to wait for a variable length of time prior to becoming member of the DHT. Now, by establishing DHT in a per-zone manner, the DHT establishment process is now physically distributed, which means nodes may become members of a DHT through the (distributed) instantiation nodes that are physically much nearer to themselves; hence the joining process is more efficient. Establishing a DHT per zone replicates another scenario in the real network: that DHTs of individual merging networks must also be merged to enable sharing of distributed data. Consider zones as medium size networks. When these networks move to close proximity, and compose together to form larger networks, their DHTs should be merged too. The bridging between established DHTs in different zones is therefore the technique that to be used to enable individual networks/DHTs that come to close proximity will compose and share distributed data. Note that when deploying DHT bootstrapping in medium size networks, it is not necessary to divide the network into zones (each medium size network itself is equivalent to a zone). Network division is needed only should the network is large. As such, our solution is provisioned to handle DHT deployment in both mediums to large size, dynamically moving and composing networks.

B. Routing Locality, Zone Division & Zone Head Election

We have mentioned that routing locality must be optimised in the resultant established DHT. Routing locality can be optimised by maintaining a coherent mapping between overlay neighbourhood and underlying routing neighbourhood [4][5]. As explained, in our previous work [8], we have developed the absorption approach to optimise DHT routing locality from this perspective. In this paper, we investigate alternative approaches to optimise routing locality: by requiring DHTs to be established per zone, of which nodes are physically near to each other, hence reduces routing delay. Routing locality can also be optimised by using a more efficient routing protocol, that routes packet to their destinations quicker (or with less overhead). There are several routing protocols available. In [10], a comparison was conducted to compare existing routing protocols such as the Max-Min Multi-hop routing protocol (MMMH) [9], Dynamic Source Routing (DSR) [11] and Ad-hoc On-demand Distance Vector (AODV) [12] in wireless ad-hoc environments. The results show that MMMH outperforms other protocols in terms of end-to-end delay and routing load at any pause times. On the other hand, in a very dynamic environment (e.g. at pause time 0), AODV performs better in delivering packets to their destination; whereas at medium to low pause time, the packet delivery ratio of the three protocols are similar. MMMH or DSR are more suitable than AODV for our solution (in terms of packet delivery, end-to-end delay, and routing load): although AODV has a better packet delivery ratio in a very dynamic environment, but we have discussed that it is not logical for a node with high mobility to be part of a distributed data sharing architecture.

A hierarchical routing protocol also fits better to our design. We have mentioned that the size of the zones depends on the size of the entire network. The number of zones should increase if the size of the entire network is large. One challenge in DHT bootstrapping is therefore to divide a network into zones without incurring additional undesirable overhead. Furthermore, once zones are created, there is a need to elect zone heads. Since zones are divided based on physical network topology, we envisage that there are possibilities of dividing a network into zones by utilising underlying physical routing information. Here, we present an elegant way to divide the
network into zones, by adopting the key concepts of MMMH. The basic idea of MMMH is to divide a network into clusters, with cluster heads managing routing between nodes within their own clusters, and routing between cluster heads themselves. Nodes in a cluster are $d$ hops away from their cluster heads, where $d \geq 1$. We chose MMMH as the underlying routing protocol for DHT bootstrapping for two reasons: it has advantages over other protocols in terms of end-to-end delay and routing load [10], and it creates a hierarchical structure which is advantageous for creating zones and electing zone heads (see shortly later).

To divide a network into zones, we do not need the exact size of the network, but a reasonable estimation would allow us to determine the size of each zone. A hierarchical routing protocol such as MMMH could provide the necessary information for this estimation: suppose we have a total of $n$ cluster heads in the network, and each cluster head maintains nodes that are $d$ hops away. We assumed all links are ad-hoc that has a maximum radius of, say, $x$ metres. Thus, assuming 2-dimensional coverage, the maximum circular coverage of one cluster head is $\pi (x * d)^2$. Suppose each wireless ad-hoc node needs a space of $s$ m$^2$, the number of nodes covered by one cluster head would be $[\pi (x * d)^2]/s$. An estimation of the total number of nodes in the network is therefore $n*[\pi (x * d)^2]/s$.

Once the size of the entire network is determined (estimated), we can figure out how many zones are needed$^2$, hence the number of nodes in each zone. Each cluster head communicate with other cluster heads that are physically nearest to itself to form zones of the desirable size. Once zones are created, cluster heads of a zone automatically become zone heads. For those zone heads that are also member nodes, they will elect a representative to be the instantiation node that instantiates the DHT establishment process, and so on. Note that zone division and zone head election are therefore integrated with the underlying routing protocol to minimise the additional overhead on setting up zones and electing zone heads (more specifically, electing the DHT instantiation node). Because each zone head is equally responsible for mapping searches between assorted DHTs, mappings are therefore carried out in a distributed manner in the sense that no one node in the zone is permanently responsible for carrying out mapping. If a cluster head drops out (e.g. run out of power, left the network...), the replacing cluster head determined by the MMMH protocol automatically becomes the new zone head. As such, automatic zone election in a dynamically changing network is achieved.

C. Securing the Protocol

The communications between the peers in DHT negotiation must be secured, which generally requires the use of symmetric keys. However, due to the dynamic nature of wireless ad-hoc networks, pre-established symmetric keys cannot be assumed on nodes. A dynamic peer-to-peer key establishment protocol is presented here. For more detail, readers are referred to [18]. The protocol involves three message exchanges: INIT, AUTH, and COMPLETE. The INIT message is sent by the sender to the receiver. It contains the sender’s Diffie-Hellman (D-H) public value, nonce, and an authenticator. The receiver sends the AUTH message, which contains the receiver’s D-H public values, nonce, and authenticator. The sender sends the last message (i.e. COMPLETE), which contains a countersigned receiver’s nonce.

V. Evaluation

A. Reduced Overhead for DHT Negotiation

The advantage of DHT bootstrapping is that it enables more rapid real-time negotiation between potential member nodes. In this evaluation, we monitor the total number of messages that were exchanged to complete DHT negotiation(s) in networks of different sizes under different approaches. Two approaches were evaluated: first, potential member nodes broadcast their DHT preferences in a network-wide environment; second, nodes carry out per-zone negotiation using DHT bootstrapping. In the latter approach, the network is split into 50 zones. Each zone head then queries each of the potential member nodes within its zone individually, and each potential member node responds with its own preference on characteristics of the to-be-established DHT. To simulate worse case scenario, we assume one zone head per zone (i.e. centralised negotiation). Figure 3 shows the results. The results show that DHT bootstrapping is much more scalable and efficient. This is because DHT bootstrapping organises nodes into zones, and requires negotiations to be conducted within zones, rather than carrying out a network-wide negotiation. Negotiations in one sense are carried out in a distributed fashion (i.e. distributed in a per-zone basis).

B. Reduced Overhead for DHT Establishment

In this evaluation, we determine the number of message exchanged to complete a DHT merging between two networks/DHTs. We use Chord-DHTs in this evaluation. Two networks are merged. The merging is between a 16-node DHT with DHTs of different number of nodes. We fixed the size of one of the merging DHTs to replicate real-life scenario (that networks may merge with others of different sizes). We assume one zone head per 16 nodes in the zone/network when bridging. Because we assume a linear physical network topology, we further assume that only one node in each

\[ \text{No. of message exchanged} \]

\[ \text{Total number of participating nodes} \]

- **Figure 3. Overhead between different approaches**

2 In general, more zones would be needed for larger networks (i.e. more zones means more distributed DHT negotiation and establishment).
network sees each other (i.e., one pair of nodes are within physical range). When discarding networks/DHTs, we always discard the networks/DHTs with smaller number of nodes. Figure 4 shows the results. The x-axis shows the number of nodes in one of the merging DHTs (the other DHT has a fixed size of 16 nodes). The y-axis shows the number of messages exchanged to complete the merging process. The results show that DHT bridging creates much less overhead than merging between DHTs. This is because bridging does not modify the original DHT keyspace structures of the merging DHTs, thus much less overhead is needed to accommodate nodes from the discarded DHT. The only overhead is to notify zone heads of nearby zones/networks/DHTs. As such, the overhead is much lower.

VI. CONCLUSION

Existing DHT implementations assume that large number of wireless ad-hoc nodes would come together at one point and at one place, and would all agree on the characteristics of the to-be-established DHT. Little effort has been made on investigating how this agreement can be reached in real-time. This paper presents DHT bootstrapping, which is dedicatedly designed to enable secure, efficient, and scalable DHT deployment in wireless ad-hoc networks of different sizes. DHT bootstrapping enables scalable and efficient real-time negotiation between nodes to come to a DHT agreement; also, pending nodes and new joining nodes are accommodated. Our solution provisions for security, and takes into account node heterogeneity, dynamicity, and scalability.

ACKNOWLEDGEMENT

This paper describes work undertaken in the context of the EU-funded IST Ambient Networks (Phase 2) project.

REFERENCES


