A Reliable Topology for Efficient Key Distribution in Ad-Hoc Networks
(Invited Paper)

Kenji Miyao[†*], Hidehisa Nakayama[†], Nirwan Ansari[‡], Yoshiaki Nemoto[†], and Nei Kato[†]
† Graduate School of Information Sciences, Tohoku University, Sendai, Japan
‡ Advanced Networking Lab., ECE Department, New Jersey Institute of Technology, Newark, NJ, USA
*miyao@it.ecei.tohoku.ac.jp

Abstract

Data confidentiality is one of the most important concerns in security of ad-hoc networks which have been widely studied in recent years. In this paper, we consider the public-key cryptography which is one of the simplest and viable means to maintain data confidentiality. There are several ways to distribute a public key. Flooding is an intuitive approach to distribute each node's public key. However, the normal flooding approach is costly, and can cause MAC-level contention in a dense region of nodes. Tree based topology flooding can be appied to mitigate these problems. The construction algorithm should use ideally only local information. In this paper, we propose a completely localized algorithm called the Local Tree-based Reliable Topology (LTRT) algorithm, which achieves both reliability and efficiency. LTRT is a localized version of TRT that has 2*-edge connectivity. Each node can distribute its public key to all other nodes in the network by LTRT. Simulation results show the efficiency of LTRT and its superiority over other localized algorithms.*

1 Introduction

Recent advances in wireless and mobile technologies have fostered the development of ad-hoc networks. Security of ad-hoc networks remains a major concern. Cryptography is a viable means to secure ad-hoc networks, but how to distribute the key remains a challenging issue.

The private-key distribution is an attractive means to maintain data confidentiality. However, if a malicious node acquires the key, the purpose of cryptography falters, and hence several protocols have been proposed to secure distribution of private keys. For example, distributed key agreement protocols have been used for secure sharing of private keys [1]. On the other hand, public-key distribution is simpler because the key is used for encryption, but not for decryption. To certify the validity of the participating node from the key, the network must have a certificate authority (CA), which however defeats the very nature of ad-hoc networks.

In some cases, public-key cryptography without any certification can fulfill certain security demand. So, the major emphasis for consideration is how to distribute the public key in such cases. When each node knows the path to any other node, an efficient broadcast scheme can be applied. Since the key distribution is always the first operation, a node cannot know *a priori* the topology of the network. Hence, the most viable means is flooding. Since flooding incurs high cost as mentioned earlier, this simple implementation may lead to tremendous energy consumption, and is thus not suitable for ad-hoc networks because the battery lives of the nodes are limited. To reduce the flooding cost, many topology control algorithms have been proposed. These algorithms are generally localized, i.e., each node uses only the information that is one-hop away. Among the localized algorithms, the most cost efficient topology is the tree based topology. One such topology is the Local Minimum Spanning Tree (LMST) [2] which can be obtained by an MST-based localized algorithm. Although LMST is cost-efficient, there is almost always only one fixed path between every pair of nodes. So if there is a link failure, the network will be split and some nodes will not be able to get the distributed key.

In this paper, we propose the Local Tree-based Reliable Topology (LTRT) algorithm which is motivated by LMST and the Tree-based Reliable Topology (TRT) [3]. LTRT is a localized version of TRT which guarantees 2-edge connectivity. The objective is to acquire an efficient and reliable topology for flooding-based key distribution by using LTRT.

The remainder of this paper is organized as follows. Section 2 reviews some related work. A brief overview of existing algorithms, LMST and TRT, are given in Section 3. In Section 4, we introduce the LTRT algorithm and discuss the properties of LTRT. Performance comparisons of LTRT

978-1-4244-2100-8/08/$25.00

with other existing algorithms are illustrated in Section 5. Finally, Section 6 concludes the paper and presents the future works.

2 Related works

Here, we briefly describe related works. There have been a rich literature on public key management.

Zhou and Haas [4] proposed the method by using threshold cryptography to distribute the CA functionality to several nodes. Some specific nodes, called servers, store public keys of all the nodes in the network, as well as are aware of the public keys of other servers. They can establish secure links among them. In [5], the authors proposed a trust model for ad-hoc networks without an CA. In this model, each node signs certificates for other nodes. Capkun et al. [6] proposed a fully distributed self-organizing public key management scheme.

In this paper, we focus on distributing the public key in ad-hoc networks. We assume that once the public keys have been successfully distributed, data confidentiality can be guaranteed by cryptography.

3 Existing algorithms

In this section, we first review the LMST algorithm. Since LMST is a 1-edge connected network, it has some reliability deficiency. In order to apply the concept of TRT to LMST in the next section, we will also briefly summarize TRT.

3.1 LMST: Local Minimum Spanning Tree

LMST is a "localized" algorithm to construct MST based topology in ad-hoc networks by using only information of the nodes which are one-hop away. Every node knows its position by GPS and has its ID for identification. The idea of LMST is simple. Each node calculates MST independently from the information of one-hop nodes and only keeps one-hop on-tree nodes as neighbors.

The procedure of constructing LMST is composed of two phases. First, each node broadcasts a "Hello" message which contains its ID and position by the maximal transmission power and obtain the information of the one-hop nodes. Each node then has its local graph in this phase. In the next stage, each node applies Prim's algorithm independently to obtain its local MST and keeps its on-tree one-hop nodes as its neighbors.

If every link has a unique weight, (i.e., different links have different weights), the locally calculated MST is also unique and the connectivity can be guaranteed [2]. The topology of LMST may not be a spanning tree but may have some redundant edges. LMST has several noteworthy features. The node degree of any node is bounded by 6, that can help reduce MAC-level contention and interference. The resulting topology can be converted into the one with only bi-directional links by removing all uni-directional links.

Note that the topology of the resulting LMST might be split by a single link failure. This might limit its applicability since the topology in an ad-hoc network should have some redundancy because of its unsure links.

3.2 TRT: Tree-based Reliable Topology

In [3], a 2-edge connected network as Reliable Topology (RT) and an algorithm to construct such a network by combination of spanning trees have been proposed.

TRT is constructed as follows. Given $G(V, N)$, a connected network topology. First, calculate one of its spanning trees, $T(N, \hat{E})$. Remove all the links in $\hat{E}$ from $G(V, N)$, and denote the rest of network as $G(N, E - \hat{E})$ that consists of n $(n \geq 1)$ connected sub-networks which are $G_1(N_1, E_1)$,$G_2(N_2, E_2)$,...,$G_n(N_n, E_n)$. Calculate $T_1(N_1, \hat{E}_1)$,$T_2(N_2, \hat{E}_2)$,...,$T_n(N_n, \hat{E}_n)$ which are the spanning trees of $G_1(N_1, E_1)$, $G_2(N_2, E_2)$,...,$G_n(N_n, E_n)$, respectively. The topology $D(N, \tilde{E})$ constructed by combining $T(N, \hat{E}), T_1(N_1, \hat{E}_1), T_2(N_2, \hat{E}_2), \ldots, T_n(N_n, \hat{E}_n)$ is referred to as a Tree-based Reliable Topology (TRT).

The construction procedure of TRT is illustrated by an example as shown in Figure 1. Given a network $G(N, E)$ as shown in Figure 1, we can construct one of its TRTs, $D(N, \tilde{E})$, by combining $T(N, \hat{E})$, $T_1(N_1, \hat{E}_1)$, and $T_2(N_2, \hat{E}_2)$, where $T(N, \hat{E})$ is one of the spanning trees of $G(N, E)$, and $T_1(N_1, \hat{E}_1)$ and $T_2(N_2, \hat{E}_2)$ are the spanning trees of $G_1(N_1, E_1)$ and $G_2(N_2, E_2)$, respectively, which are the remaining networks after removing the links in $T(N, \hat{E})$ from $G(N, E)$.

It can be observed that if $n = 1$, $G(N, E - \hat{E})$ is still a connected network, TRT is actually constructed by combining the two spanning trees of $G(N, E)$. Authors in Reference [3] suggested to deploy minimum spanning tree (MST) to construct TRT, i.e., in the process of constructing the TRT, all the spanning trees are the minimum spanning trees of the corresponding networks.

4 LTRT: Local TRT

In this section, we propose LTRT which is a localized version of TRT. LTRT has the same reliability as TRT, i.e., LTRT is also a 2-edge connected network if the original network is also 2-edge connected if the original network is i-edge connected, where $i \geq 2$. Also, the time complexity is small.

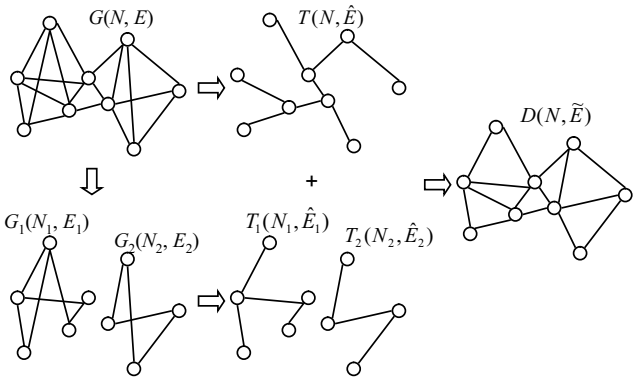

Figure 1. Construction of TRT

To simplify the discussion, consider a network in which every node has the same transmission range r_{max}. Let the network topology be represented by an undirected graph $G = (N, E)$, where $N = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes and E is the set of links in the network G. A unique ID is assigned for each node and each node can obtain its own position in the network by GPS or a lightweight localization technique for wireless networks. This position information is only used for calculating the link cost between every two nodes, and it is not necessary if the link cost can be calculated by reception of the broadcast. For illustrative purposes, we assume every node uses GPS to acquire its own position.

4.1 Construction of LTRT

LTRT can be easily constructed by applying the topology construction phase of LMST two times. The procedure of constructing LTRT is composed of four phases: information exchange, first topology construction, link deletion, and second topology construction.

1) Information Exchange: Each node broadcasts a "Hello" message which contains its ID and position, and obtains the information of its one-hop nodes.Each node obtains its local graph $G_u(N_u, E_u)$ in this phase.

2) First Topology Construction: Each node u applies Prim's algorithm independently to obtain its local MST $T_u^1(N_u, E_u^{1'})$ and broadcast neighbors $N'(u) = \{v|(u, v) \in E_u^{1'}\}$. Each node u can obtain all neighbors of $v \in N_u$ from the broadcast, and then it can fix the first topology $D_u^1(N_u, E_u^1)$. $D_u^1(N_u, E_u^1)$ is an undirected graph constructed by eliminating unidirectional edges. This elimination of unidirectional edges does not destruct the connectivity of the network.

3) Links deletion: Each node u deletes the links in E_u^1 from its local graph $G_u(N_u, E_u)$, resulting in the topology, $G_u^2(N_u, E_u - E_u^1)$.

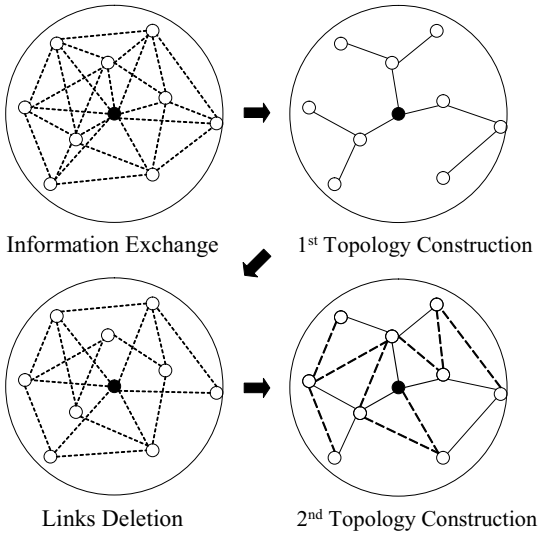

Figure 2. Operations of each phase

4) Second Topology Construction: Each node u applies Prim's algorithm to $G_u^2(N_u, E_u - E_u^1)$ independently to obtain its second local MST $T_u^2(N_u, E_u^{2'})$ and broadcast neighbors $N'(v) = \{v|(u, v) \in E_u^2\}$. The local graph $G_u^2(N_u, E_u - E_u^1)$ may be separated in n graphs, but it is sufficient to apply the Prim's algorithm to one of the graphs as long as it includes node u because the necessary information is only the neighbor set of u and broadcast neighbors $N'(u) = \{v|(u, v) \in E_u^{2'}\}$. Each node then can fix the second topology $D_u^2(N_u, E_u^2)$ from its broadcast neighbors by including only bidirectional edges. Finally, each node u can decide its neighbor set $N(u) = \{v|(u, v) \in E_u^1\} \cup \{v|(u, v) \in E_u^2\}$.

There are some noteworthy features in LTRT. LTRT is an undirected graph. The bulk of the computation is due to the cost of Prim's algorithm, and so the computational cost is very small. Only three broadcasts per node are required, i.e., low overhead. If the weight of any link has the unique value, the choice on the minimum weight edge e is unique, and thus the topology calculated by Prim's algorithm is unique; therefore we can always obtain unique topology from LTRT.

When nodes join or leave the network, the topology will be recalculated. However, it is not necessary for all the nodes of the network to recalculate its topology because the algorithm is localized. In such a situation, the nodes that are one-hop away from the joining/leaving node start phase 2), and nodes that are two-hop away proceed to phase 3) and 4). Hence, only nodes that are two-hop away or less from the joining/leaving node need to recompute their topologies when the network is changed.

4.2 2-edge connectivity

The topologies constructed in phase 2) or 4) are the same as LMST. The connectivity of either of the resulting topologies is proved in [2]. It can be similarly shown that LTRT has 2-edge connectivity.

Theorem 1: Given a 2-edge connected network $G(N, E)$, the LMST $T(N, \hat{E})$, constructed in phase 2), LMSTs, $T_1(N_1, \hat{E}_1)$,$T_2(N_2, \hat{E}_2)$,...,$T_n(N_n, \hat{E}_n)$, constructed in phase 4), and the LTRT, $D(N, \tilde{E})$, constructed by combining $T(N, \hat{E})$, $T_1(N_1, \hat{E}_1)$, $T_2(N_2, \hat{E}_2)$,...,$T_n(N_n, \hat{E}_n)$, LTRT $D(N, \tilde{E})$ is also 2-edge connected.

Proof: We prove this by showing any edge $\{e\}$ is not an edge-cut of $D(N, \tilde{E})$. Assume an edge $\{e\}$ is an edge-cut of $D(N, \tilde{E})$ that splits $D(N, \tilde{E})$ into $D_1(N_1, \tilde{E}_1)$ and $D_2(N_2, \tilde{E}_2)$ by removing $\{e\}$, but does not split $G(N, E)$. $e \in \hat{E}$ is obvious because $T(N, \hat{E})$ is connecting all the nodes and $T(N, \hat{E}) \subset D(N, \tilde{E})$. However, there are some edges $\{e_1, e_2, \ldots, e_n\} \in (E - \hat{E})$ that connect between the set of N_1 and N_2 but are not the links of $\tilde{E}$, since $G(N, E)$ is 2-edge connected. Since the topologies constructed in the second topology construction phase are connected, one of $e_1, e_2, \ldots, e_n$ must be in one of $\hat{E}_1, \hat{E}_2, \ldots, \hat{E}_n$. This, however, contradicts our assumption that $\{e\}$ is the edge-cut of $D(N, \tilde{E})$. Hence, none of the edges $\{e\}$ is an edge-cut of $D(N, \tilde{E})$, that is, $D(N, \tilde{E})$ is 2-edge connected. ■

In the simplest term, TRT is composed of spanning trees, and LTRT is composed of topologies that include spanning trees with some redundant edges. So LTRT is composed of TRT with some redundant edges, that end up to have the same reliability. This also shows that if the graph constructed in phase 2) and 4) is connected, the topology is always 2-edge connected.

4.3 Complexity analysis

We show the time complexity of LTRT construction is $O(n \log m)$, where n is the number of nodes which are one-hop away and m is the number of links in the local network $G_u(N, E)$. It is the same as LMST.

In the information exchange phase, each node broadcasts and obtains the information. In this phase, adding the neighbors in the local graph costs $O(n)$. Each node calculates the length from the node positions to obtain link lengths in its local graph, thus costing $O(n^2)$. If each node u calculates only the link lengths between the neighbor node v and u and broadcast its length, the cost can be lowered to $O(n)$. In the first topology construction phase, each node applies Prim's algorithm with complexity of $O(m \log n)$. If we employ Fibonacci heap, the complexity is $O(m + n \log n)$. In the links deletion phase, the deletion of a link from the network is $O(\log n)$ because there are a maximum of n links for each node. Since the node degree of any node is bounded by 6, the number of manipulations is $O(n)$. So, the deletion cost is $O(n \log n)$. In the second topology construction phase, the time complexity is, like the first topology construction, $O(m \log n)$, or $O(m+n \log n)$ if Fibonacci heap is adopted.

As mentioned above, the minimum time complexity is $O(m + n \log n)$. Since the cost of calculation is almost that of Prim's algorithm, the actual computational complexity is rather low, and the algorithm can be easily applied.

5 Performance evaluation

Other localized algorithms have been proposed to construct a 2-connected topology, but most of them have 2-vertex connectivity which has stronger connectivity than 2-edge connectivity. If a node is dropped, the topology has to be recalculated, thus incurring large computational complexity. Therefore, it is sufficient to employ 2-edge connectivity, a weaker one, which leads to a low cost topology.

CBTC(α) [7] is 2-vertex connected if $\alpha \leq \frac{\pi}{3}$. The connectivity is proved in [8]. Fault-tolerant Local Spanning Subgraph (FLSS$_k$) [9] is k-vertex connected. It uses a greedy algorithm, and so the topology of FLSS is nearly optimal. It was shown that the computational complexity is $O(m)$ if $k \leq 3$. However, this does not include the sorting cost of links which is $O(m \log m)$, and the actual computational complexity is much higher than that of LTRT.

In order to understand the effectiveness of our algorithm, we evaluate the performance of LTRT against TRT, LMST, CBTC($\frac{\pi}{3}$), and FLSS$_2$ algorithms via extensive simulations. We generated a network where 100 nodes are randomly placed in a square region. Each node has a maximum transmission radius of $250[\mathrm{m}]$. The length of the square region is $1000[\mathrm{m}]$.

Figure 3 shows the topologies after applying respective algorithms in the square 1000[m]. LTRT, the localized TRT, acquires nearly the same topology as TRT. It can be found that the topologies of both FLSS$_2$ and LTRT are very close.

Table 1 shows the average node degree, average radius, and average link length. The node degree is defined as the number of neighbors of each node and is an indication of

Table 1. Simulation Result

Algorithm	Degree	Radius	Link Length
NONE	16.70	250.00	162.24
LMST	2.05	88.67	70.38
CBTC	12.82	222.19	149.00
FLSS	3.88	126.53	88.91
TRT	3.96	121.88	86.86
LTRT	4.11	124.26	88.64

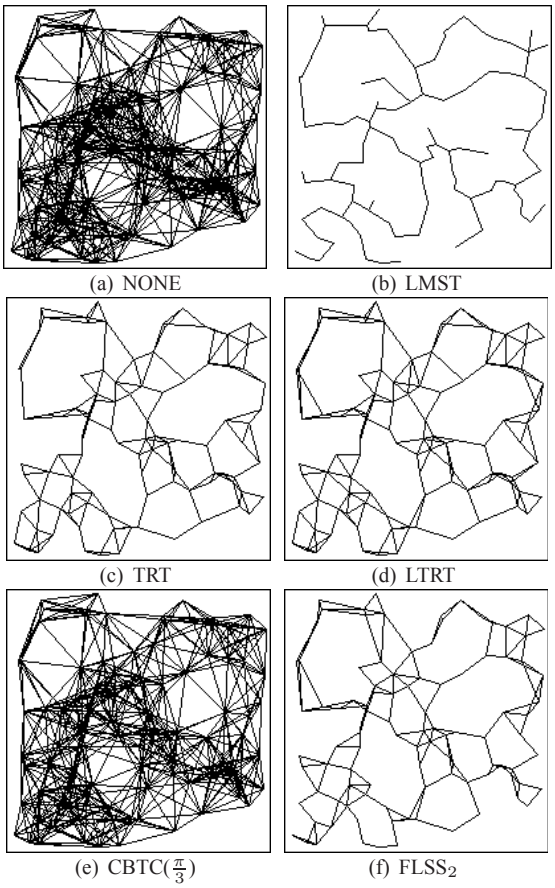

Figure 3. The topologies derived under each algorithm

the level of MAC interference. The average link length and radius are directly linked to the flooding cost and energy consumption. These values are obtained by averaging over 50 runs of simulations. Since LMST has only one-connectivity, it yields small value in each measure. LTRT outperforms CBTC($\frac{\pi}{3}$), and it is almost the same as FLSS$_2$ which is 2-vertex connected and near the optimal. Since the computational cost of LTRT is much lower than that of FLSS$_2$, this simulation shows that LTRT achieves comparable performance as that of FLSS$_2$ but at a much lower computational cost.

6 Conclusion and future works

Key distribution is one of the most important concerns in secure ad-hoc networks. In this paper, we have proposed the LTRT algorithm which is a localized version of TRT for efficient and reliable key distribution by flooding. We have evaluated the performance of LTRT through extensive simulations, and showed that LTRT achieves comparable performance as that of the near-optimal algorithm, but at a much lower computational cost.

Our future work will further evaluate LTRT in more complex scenarios by varying the node density and size of network. We will next generalize LTRT to have k-edge connectivity since k-edge connectivity may enhance the reliability in networks with more unsettled link state.

Acknowledgements

This work has been funded in part by Strategic International Cooperative Program, Japan Science and Technology Agency (JST), National Science Foundation Cyber Trust under grant no. 0726549, and Sendai Advanced Preventive Health Care Services Cluster from the Ministry of Education, Culture, Sports, Science and Technology.

References

[1] M. Steiner, G. Tsudik, and M. Waidner, "CLIQUES: A new approach to group key agreement," in *Proc. of the 18th International Conference on Distributed Computing Systems*, May 1998, pp. 380–387.

[2] N. Li, J. Hou, C. Sha, and L. Sha, "Design and analysis of an mst-based topology control algorithm," *IEEE Trans. on Wireless Communications*, vol. 4, no. 3, pp. 1195–1206, May 2005.

[3] N. Ansari, G. Cheng, and R. Krishnan, "Efficient and reliable link state information dissemination," *IEEE Communications Letters*, vol. 8, no. 5, pp. 317–319, May 2004.

[4] L. Zhou and Z. Haas, "Securing ad hoc networks," *IEEE Network*, vol. 13, no. 6, pp. 24–30, Nov.-Dec. 1999.

[5] J. Hubaux, L. Buttyán, and S. Capkun, "The quest for security in mobile ad hoc networks," in *Proc. of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, Oct. 2001, pp. 146–155.

[6] S. Capkun, L. Buttyan, and J. Hubaux, "Self-organized public-key management for mobile ad hoc networks," *IEEE Trans. on Mobile Computing*, vol. 2, no. 1, pp. 52–64, Jan.-Mar. 2003.

[7] L. Li, J. Y. Halpern, P. Bahl, Y. Wang, and R. Wattenhofer, "Analysis of a cone-based distributed topology control algorithm for wireless multi-hop networks," in *Proc. ACM Symposium on Principles of Distributed Computing*, Aug. 2001, pp. 264–273.

[8] M. Bahramgiri, M. Hajiaghayi, and V. Mirrokni, "Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks," in *Proc. of the IEEE Int'l. Conf. on Computer Communications and Networks*, Oct. 2002, pp. 392–397.

[9] N. Li and J. Hou, "FLSS: a fault-tolerant topology control algorithm for wireless networks," in *Proc. of the 10th annual international conference on Mobile computing and networking*, Sept. 2004, pp. 275–286.