A Robust, Distributed TGDH-based Scheme for Secure Group Communications in MANET

Maria Striki, John S. Baras and Kyriakos Manousakis
Institute for Systems Research and
Electrical and Computer Engineering Department
University of Maryland, College Park
MD 20742, USA

***Abstract*—Securing multicast communications in Mobile Ad Hoc Networks (MANET) is considered among the most challenging research directions in the areas of wireless networking and security. MANET are emerging as the desired environment for an increasing number of commercial and military applications, addressing also a growing number of users. Security on the other hand, is now an indispensable requirement for these applications. However, the limitations of the dynamic, infrastructure-less nature of MANET impose major difficulties in establishing a secure framework suitable for such services. The design of efficient key management (KM) schemes for MANET is of paramount importance, since the performance of the KM functions imposes an upper limit on the efficiency and scalability of the whole secure group communication system. In this work, we contribute towards this direction by extending TGDH to a novel distributed scheme: DS-TGDH. Our aim is to modify TGDH to: *a*) be feasible in the most general resource-constrained MANET where no nodes with special capabilities exist, *b*) produce considerably lower overhead for the network nodes involved, *c*) handle disruptions with low cost. We consider the underlying routing protocol in our design, and we apply a distributed TGDH version over a robust schedule, optimizing parameters of interest. We focus on the design and analysis of the "stealthy" TGDH and compare it with the original. Through our analysis and results we shed insight on the actual feasibility of these protocols for MANET and provide realistic and fair comparison results that more accurately advocate the pros and cons of each protocol over the environment of study.**

I. INTRODUCTION

A Mobile Ad Hoc Network (MANET) is a collection of wireless mobile nodes, communicating among themselves over possibly multi-hop paths, without the help of any infrastructure such as base stations or access points. As the development of wireless multicast services such as cable TV, secure audio and conferencing, military command and control grows, the research on security for wireless multicast becomes increasingly important. The role of key management (KM) is to ensure that only valid members have access to a valid group key at any time. So, the existence of a secure, robust KM scheme for multicast communications is essential. However, the characteristics of MANET constitute the major challenge for the design of such schemes. We are dealing with dynamic, infrastructure-less networks of limited bandwidth, unreliable channels, where topology is changing fast. Network nodes may have limited capacity, processing and transmission power. Connections are temporary (mobility, battery drainage) and unreliable. These constraints render existing schemes inefficient in MANET: among other requirements, they need to catch up with a rapidly changing topology, and handle failures at any time during group key establishment.

Along with the requirement to design secure KM schemes that achieve better performance than existing ones (either for wire-line or wireless networks), the need for the KM schemes to handle successfully, and tolerate with low impact, network dynamics and failures (robustness) in a network with large number of nodes (scalability) is now equally important.

In an attempt to meet all these objectives, two novel Octopus schemes: *Modified Octopus* MO and *Modified Octopus with Tree* MOT [6] were introduced and evaluated in addition to the original [1]. The special features of Octopus schemes have motivated our interest to explore and extend them. Hierarchy is supported through the partition of a large key agreement (KA) group to 2^d smaller subgroups. The superiority of MOT has been mainly attributed to the incorporation of Tree Group Diffie Hellman (TGDH).

The primary focus of our earlier work was the analysis of the proposed schemes, based on the overhead resulting from the processing and exchange of KM information only. That was done in isolation from underlying backbone network functions (i.e. routing, clustering, leader election). However, since each of the schemes considered relies upon such functions in a different way, our previous evaluations are not totally fair. The consideration of parameters that can accentuate their individual traits and the differences in their performance will provide more realistic results. For example, the hierarchical framework of the three schemes is supported via a clustering mechanism. Although the cost of adding and maintaining it adds to the overall cost of each individual scheme, it does not alter the outcome of their comparative evaluation. On the contrary, the communication schedule within the subgroup differs: scheduling is not required for the centralized (O), in contrast to MO and MOT, where members interact among themselves for the subgroup key generation (KG). A simple schedule based arbitrarily on members' *ID*s is likely to result in unnecessary routing. The need to execute the subgroup schemes in MO and MOT over a schedule that optimizes metrics of interest comes into play. The network assumptions made in TGDH [4], (e.g. leader reaches all members via a single broadcast), are too simplistic to apply to a general MANET, but mitigate the need to consider the functions referred to above. In real multi-hop networks, a holistic consideration proves essential for a concrete KM framework. It would be worth exploring if MOT still prevails after a re-evaluation of the same schemes under more realistic network assumptions.

In the present work, we assume that a path between group members may as well include non-member relays. We rely on the routing redundancy to ensure that the exchanged messages are delivered in a timely manner. Dividing such group into

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

subgroups, each corresponding to a complete members graph, reachable by the subgroup leader via a single broadcast, would permit the execution of the original TGDH exactly under the assumptions described in [4]. This approach is impractical for a large group: a high number of subgroups will be formed, sensitive to even subtle mobility changes, and it is quite likely that they will contain very few members, even a single one. Even subgroups deployed in close proximity still cannot be merged. The result is a considerable waste in network resources, and an infeasible execution of Hypercube.

In this paper we present an adaptation of TGDH to meet the requirements of a general MANET. In particular, we modify TGDH so that: *a*) it is made distributed, there is no single point of failure leader, *b*) it is executed on a schedule that optimizes our own defined routing and robustness metrics, under a topologically aware consideration, *c*) it tolerates failures and disruptions with low cost, *d*) it is far more efficient w.r.t. bandwidth and computation overhead. We denote this novel scheme as **DS-TGDH** (*Distributed TGDH with Schedule*) and evaluate both protocols under the new assumptions. Section 2 gives an overview of related work. Section 3 provides an overview of TGDH. Section 4 discusses fault-tolerance issues for our framework. In section 5 we introduce DS-TGDH, and in section 6 we analyze its initial and steady state operations. Sections 7, 8 present the analytical comparative performance evaluation and the corresponding simulation results. Finally, in section 9 we conclude the paper.

II. Related Work

The KM proposals for secure group communications abound in the literature. From the perspective of **contributory** protocols (equal member contributions for the group KG), Becker et al. [1], derived lower bounds for the gossip problem and proved them realistic for Diffie-Hellman (DH) protocols. They used the basic DH distribution extended to groups (Steiner [2]). GDH.2 is the most efficient representative, minimizing the total number of message exchanges. **TGDH** by Kim et al. [10], is a hybrid, efficient protocol that blends binary trees with DH key exchanges. Becker in [1], introduced Hypercube that requires a minimum number of rounds, and in [5], Asokan added limited fault-tolerant extensions. Becker introduced Octopus that requires minimum number of messages and then 2^d-Octopus combining Hypercube with Octopus to a very efficient scheme for any number of nodes.

Centralized protocols are based on a simple key distribution center. The most fundamental representative is GKMP [9], in which a group leader shares a secret key with each member and uses it to send the group key to its members. LKH [8], creates a hierarchy of keys for each member. Each group member is secretly given one of the keys at the bottom of the hierarchy and can decrypt the keys along its path to the root. Evolution of the latter are: ELK [20], designed rather for a stationary network, and OFT [7] that minimizes the number of bits broadcast to members after a membership change.

Some more recent proposals exist for wireless ad-hoc networks. Even these schemes, do not seem to scale well or handle successfully the network dynamics. Some of these approaches rely on public key cryptography, which is very expensive for resource constrained nodes, or on threshold cryptography [14, 15, 16, 21], which results in high bandwidth, does not scale well, and presents security vulnerabilities, mainly due to the mobility of nodes. A different approach is based on probabilistic key pre-distribution [17, 18], which is a very lightweight method, designed for sensor networks, but has serious vulnerabilities, and requires some infrastructure to handle mobility and membership changes (revocations). Amir et al. [12, 13], focus on robust KA to make GDH protocols fault-tolerant to asynchronous network events. However, their scheme was designed for the Internet, and requires an underlying reliable group communication service and synchronization, so that preservation of virtual semantics is guaranteed. In [11], Poovedran et al. attempt to minimize the associated bandwidth w.r.t. to energy expenditure via centralized key tree schemes. The network topology is considered static, and there is no provision for adjusting the tree structure to dynamically changing networks. The optimal solution of their formulation does not scale with group size. In [6] Octopus schemes were designed to provide robust and efficient KM for group communications, but were analyzed in isolation from network functions that interact with the KM protocols.

III. TGDH Overview and Cost Evaluation

TGDH is well documented in [4]. Here, we simply address its key points. A detailed description of the cost analysis, wherever previously omitted, can be found in [6].

Overview: 2-party DHKEs are used to compute a binary tree of keys (height $h = \log_2 n$) from the leaves to the root. Each of the n members is associated with a leaf in the tree. Each tree node x is associated with two cryptographic keys, the un-blinded k_x and the blinded key (BK) $k_x' = g(k_x)$, where g is the 2-party DH function. Interior keys are defined by the rule: $k_x = g(g(k_{left(x)}), g(k_{right(x)}))$. The root key becomes the **group key**. Each member knows only the un-blinded keys on its path to the root, and the BKs of the siblings of the nodes on the same path (co-path). These BKs are sent by the leader. The authors assume that the leader (sponsor) sends all BKs to all members. A member computes the un-blinded keys along its path to the root. If one of the BKs changes and the member gets the new value, it re-computes the keys on the path and finds the new group key. The leader broadcasts to all members the new values of the BKs that change (h). Then, $2h$ exponentiations are performed by the leader, and $[1 \ldots h]$ by each member, since not all BKs change for each. For the initial tree formation, $2n$ messages are broadcast, $4n$ exponentiations are performed by the leader and h by a member. In the case of membership updates, the leader creates a new secret key for itself. In the addition case, it gets the BK of the new member and updates the corresponding path.

IV. Fault-Tolerant Extension of TGDH

We wish to design a distributed, efficient **communication schedule** on top of which TGDH can be executed, so that the combined communication-routing overhead (RC) is reduced. TGDH does not require communication regulation, since all messaging goes through the sponsor. The BK of each member at a given tree level is unicast to the sponsor who waits to

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

collect all BKs of the same level and then combines them to a single broadcast to all the rest of the members. This broadcast message signals the advancement in the tree level. Each member can now compute the designated secret value for the next level. It then blinds it and unicasts it to the sponsor. The same process is repeated for all levels up the tree until the root is reached. Hence, any two members communicate with each other via the sponsor. With this scheme, KM is executed in a centralized manner. TGDH may perform as claimed in [4] under the constraints of a limited network where any member is able to reach all group members with 1-hop and where the sponsor acquires extra bandwidth and power capabilities. Such a scenario does not reflect the general case. The most basic **drawbacks** of TGDH are the following:
(*a*) Any trusted member, with sufficient bandwidth and power capabilities should be ready to assume the duties of a leader. The most important *constraint* for MANET is the existence of such node(s). What is more, members may not be able to reach the sponsor via a single transmission anyway. So, both directions may introduce excessive routing. (*b*) Simultaneous transmissions of BKs at any tree level to the leader through multi-hop routing in a limited network area may increase the probability of collisions at the MAC layer, and consequently the probability of re-transmissions, deteriorating performance even more. (*c*) Since power is a very valuable resource, it is undesirable for members to serve frequently as relays in heavy KM messages. Also, the sponsor is burdened with heavy tasks that consume its residual energy all too fast. (*d*) The sponsor is a single point of failure communication-wise. If it fails during key establishment, TGDH is stalled until a new leader emerges, and a number of BKs must be updated.

TGDH has some **considerable advantages** as well: (*a*) it is simple, no sophisticated scheduling is required, (*b*) failure of a member has no impact on protocol other than the update of a logarithmic number of BKs, (*c*) every member knows all BKs and can proactively anticipate dynamic membership changes that would result in the need to reconfigure KG and restore the group key with minimal latency.

Considering the pros and cons of TGDH, we present a more efficient, distributed version that uses a transmission schedule algorithm to mitigate the weak points of the above approach.

V. DS-TGDH OVERVIEW

A. Transmission Schedule for DS-TGDH

The sponsor initially collects through members' registration and link state information the required information about its subgroup members. We assume **a generic** underlying **routing protocol** with the property that it always finds the minimum path (e.g. Dijkstra). It provides all nodes with the paths to at least the nearest subgroup members (w.r.t. the number of hops), and finds at least one path connecting two members, as long as both are within the same cluster. It also provides members with information of the **robustness** of neighbor paths. If we know the motion parameters of two virtual neighbors (speed, direction, radio propagation range), and their coordinates, we can determine the duration of time these nodes will remain connected, denoted as Link Expiration Time (LET) [22]. The **routing path** $\boldsymbol{r}$ is characterized by the minimum among all LETs of its links, denoted as Route Expiration Time (RET). RET indicates the overall stability of a path: as soon as a single link on a path is disconnected, the entire path is invalidated, as argued in [22]. We, as well, choose *RET* as a component of our "path robustness metric". As an additional metric to characterize path robustness we select the normalized product of the residual energies of all nodes included in the path: $EN=\frac{1}{|r|}\prod_{i\in r} E_{res}(i)$.

We do not use $min\{E_{res}(i)\}$, since a node may participate to multiple routing paths, but average the residual energies of nodes along the same routing path, and we characterize the "**robustness**" of a path r by the value: $\boldsymbol{R_r} = a\times EN + b\times RET$. The contribution of each parameter can be fine tuned through a, b. Member x maintains and updates a list L_x that includes its virtual neighbors and the collected metrics for each.

B. Execution Stage Overview

A member j that belongs to the schedule tree (ST) $\boldsymbol{T}$, selects one among the available routing paths at level (l-1) that leads to a member $k\in L_j$, and j generates now the offspring $<j, k>$ for level l. Members j and k are connected with a logical link, one of low cost w.r.t. routing, and are considered siblings at level l. They update all members in their proximity (lists $\boldsymbol{L_j}$, $\boldsymbol{L_k}$) of the new event (TreeFlag(k) is set to "*busy*" mode). Then, they individually proceed to generate offspring for level (l+1) from L_j and L_k. They decline any attempt to be enlisted in the ST by member r that did not receive the updates, and r checks its remaining options. The ST expands according to the following idea: *The more robust the members, the higher in the tree they are placed*. A member j arranges L_j w.r.t. the metrics discussed and attempts to "use" its members in this order one by one as offspring (in successive levels), unless they are already "used" by other members. Whenever a leaf is reached, information about the members that belong to that path traverses up the root. The root checks if all members are included in the ST. Tree members need only know their immediate parent, grandparent (if any) and the parent's first sibling, in addition to their own siblings. Member j gives priority to its "unselected neighbor" m that satisfies at least one of the following rules in the order they are stated:

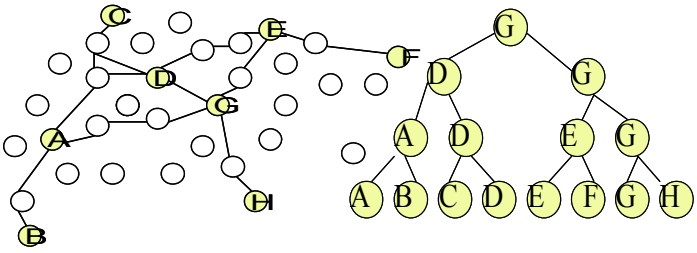
Rule#1) $m\in \{L_J \cap L_B / T\}$, where T is the current tree version, and B is either the parent or the first sibling of j. This rule ensures that if j becomes faulty then B can take its place in the tree, and become a sibling of all the previous siblings of j, so that the impact of a failure is minimal (no need to prune the tree). To ensure that the selection of m at this stage implies a relatively strong tree link also, we simultaneously impose that $R_r(j,m)>Th$, where Th is a threshold value. If there is more than one choice, the "shortest path" one is preferred.

Rule#2) $m\in \{L_j / T\}$ and $r_{jm}\le D$, s.t. $\forall\, x\in \{L_j / T\}$ and $r_{jx}\le D$: $R_r(j, m) > R_r(j, x)$. If rule#1 cannot apply, then rule#2 simply selects the member with the strongest link to j, among those with routing distance less than D hops.

In our extended version [19], we provide the pseudo-codes for offspring selection, and for member addition/eviction, to better illustrate the process.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

These rules ensure two basic requirements: *1)* Members with high risk of getting disconnected occupy the fewest possible internal nodes and are pushed towards the leaves. The impact of their "loss" is mitigated as much as possible when pruning the tree, *2)* Members high in the tree are more likely to satisfy rule#1, since they have more available neighbors. Their failure would affect the schedule of a larger subset, so it is important to anticipate and remedy such failures with low extra cost and latency. At the other extreme, failure of a member associated only with a leaf has no impact to the tree. So, a greedy strategy for selecting the offspring appears to be the best to pursue. Members, at all times, attempt to use their best available options, and keep pushing the worst candidates towards the leaves. No member is now a single point of failure. Even if the root *A* fails, the idea is that its nearest former neighbor, e.g. node *B*, replaces *A* in the tree, and routing connects to *B* all nodes prior connected to *A*. It is very likely that these nodes remain relatively close to *B* as well.



Scheme 1: TGDH schedule formation for arbitrary network graph

This algorithm does not necessarily result in a totally balanced ST. This would be desirable if emphasis was placed upon fair resource allocation. Our distributed approach indirectly achieves: ***a)*** a minimum delay ST since members are enlisted in the tree in a *FCFS* manner, and ***b)*** producing a relatively balanced ST, since at any level, all members are free to expand, if options are available. Our approach resembles mainly a *BFS* instead of a *DFS* algorithm.

VI. Analysis and Evaluation of DS-TGDH

A. Initial DS-TGDH Schedule Evaluation.

Member *j* performs $2L_j$ comparisons of cost $C_{CMP} = O(1)$ each. When *j* is handed the token, it updates L_j of the status change in its flag (*busy*). While a member *y* is being tested during the selection process by member *x*, its Flag is set to "*lock*". Those that contact *y* in the mean time cannot enlist it as long as its flag remains to "*lock*" mode. After the selection stage ends, *x* unlocks *y*'s flag by setting it either to: *busy* or *free*. The exchange of status data requires far lower number of bits (K_S) than the key data (K). We obtain the DS-TGDH processing cost for a member *j*, $C_P(j)$, under the worst case scenario:

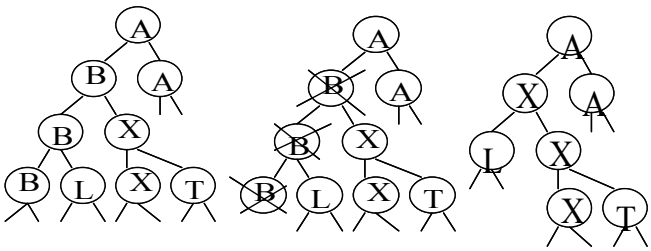
$$\boldsymbol{C_P}(j) = L_j \times (3C_{CMP} \times K_S)\ (rule1) + L_j \times (3C_{CMP} \times K_S)\ (rule2) + (3C_{CMP} \times K_S)\ (process\_results) = (2L_j+1) \times (3C_{CMP} \times K_S).$$

B. Impact of Dynamic Events on the DS - TGDH Schedule.

We want to guarantee a transmission schedule that can anticipate dynamic or membership changes as well. From the security point of view, it is shown in [4, 8] how members' evictions and additions are handled in TGDH to preserve the fundamental security properties that escort all secure KM protocols: *Forward*, *Backward* and *Group key secrecy*. Here, we want to show in addition how to resume the TGDH schedule in the event of disruptions of any kind, with the minimum amount of extra overhead and latency, and ensure that the updated schedule is still functional and efficient.

B1. Eviction:

The ST may need reconfiguration after a member's removal. The idea behind the eviction algorithm is the following: either the parent *A* or the first sibling *X* of the evicted member *B*, substitute *B* in the tree path wherever it appears. Without loss of generality, let *B* be replaced by *A*. All former siblings of *B* must now become *A*'s siblings. For members $x \in (L_B / L_A)$, routing finds the shortest paths to *A*, and they are all added to L_A. Compared to the rest of *B*'s former siblings, *A* or *X* are either more robust or lie closer to *B*. It is thus quite likely that some of *B*'s former siblings: *a)* $\in \{L_A \cup L_x\}$ already, and routing needs not generate extra paths, or *b)* $\notin \{L_A \cup L_x\}$ but the paths to be formed are relatively short, since both ends lie in the proximity of the evicted *B*. These statements most likely hold if the network is relatively dense. Under this approach, we reconfigure the ST with little latency and we expect that the new schedule does not result in substantial extra overhead.



Scheme2: Illustration of schedule maintenance after *B*'s failure, when parent *A = P(B)* is selected to "replace" *B*.

The decision about which member will substitute *B*, is assigned to the **last sibling** chosen by *B*, denoted as *L*. *L* is the least stable/most remote among the available tree neighbors of *B*. The routing finds the shortest paths from *L* to both *X* and *A*. The shortest one with robustness no worse than *Th*, designates which member will substitute *B*. The reason is that we want the substitute of *B* to accommodate all affected members with low overhead, ensuring also high robustness. It is more likely that the best selection for the least robust sibling is also the best for the rest. Ideally, we should find the shortest paths to *X* and *A* for all siblings, and select the one that accommodates the majority. The overhead and the required coordination, make this solution impractical for the network we study.

B2. Addition:

The routing finds the shortest paths from the new member *T* to members in the proximity, and a list L_T is created. *T* will be added as a leaf to the member that is connected to *T* via the minimum number of hops and fulfills a robustness criterion at the same time. Rule#1 is used to select the best solutions, otherwise *T* is attached to the member *J* with which they share the minimum hop path, for which robustness is among the highest in L_T (sub-optimal solutions). Then, *J* expands one level and becomes the parent and the first sibling of *T*.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

C. Analytical Evaluation of Dynamic Events on DS-TGDH.

Eviction/Failure: The sibling selected last by B, denoted as L, does 2 comparisons to determine B's substitute – let it be A. B's former siblings, h_B, that do not already acquire paths to A, use routing to obtain such paths and metric values. However, $h_B<|L_B|$, since a subset of L_B has been prior reserved by other tree members. Also, any $j \notin L_A$ is likely to have been enlisted by B towards the end of its path. Let N_E denote the average number of members that computes such paths. It can be shown that $N_E \approx h_B/2$. We omit the proof for lack of space.

Addition: About DM_T shortest routing paths are discovered. The total computation cost C_P for T is derived similarly to the eviction case under the worst case scenario. Let D be a limit imposed on the length of the proximity lists. Hence:

$C_P=L_T\times(4C_{CMP}\times K_S)(rule1)+L_T\times(2C_{CMP}\times K_S)(rule2)+$
$(2C_{CMP}\times K_S)$ (*process Results*) $=(3L_T+1)\times(2C_{CMP}\times K_S)$ (bits).

	DS-TGDH Adjustment Costs
Deletion Comm/tion	$N_E\times K_S$ ($> h_B/2$)
Deletion Computation	$2C_{CMP}\times K_S$
Add Comm/tion	$DM_T\times K_S$ ($<D$ paths)
Add Computation	$(3D+1)\times(2C_{COMP}\times K_S)$

VII. DS-TGDH *VS.* TGDH ANALYTICAL EVALUATION

In TGDH, the initial KG takes in h rounds. A member sends $h\times K$ bits to the leader, and receives a total of $h\times n\times K$ bits, while the useful data is contained only in $h\times K$ bits. This means that the routing path from the leader to any member carries $(n-1)\times K$ bits of "redundant" data per round, which makes this approach impractical. The total number of BKs exchanged is: $E[BK_S]=(n\times h)$ [members] + $(n^2\times h)$ [leader].

In DS-TGDH, the number of participants is reduced to half after each round. So, $2\times n$ BKs exchanges over the designated routing paths are made. Any member must get the BKs from its co-path. A member that participates actively to KG knows all the required BKs up to this point. If member A is active at step i, it has been active in all previous steps from the start of its path, and has collected and sent all the required BKs (1 per step). Let h_{UA} denote the number of times A appears in the tree ($\neq A$'s path to the root, h_A). During the upward phase, A waits to collect all keys of its co-path first, and then uses each of the h_{UA} paths to send a BK to its offspring. So, the distribution of the BKs follows a top-down approach. The number of BKs distributed from parent A to offspring B equals the number of hops B is away from the root in the ST: A receives a message $(h_A - h_{UA})\times K$ bits long and sends h_{UA} BKs to its offpring: i.e. for offspring j, at level i, A sends (h_A-i) BKs ($i \leq h_{UA}$).

Members Receive:

BKs $(A) = (h_{UA}\text{-}1)_{UP} + (h_A - (h_{UA}\text{-}1))_{DN} = h_A$.

Routes Used $\mathbf{RU_R}(A) = (h_{UA}\text{-}1)_{UP} + 1_{DN} = h_{UA}$.

$\mathbf{RU_R}$ (*Total*) $= \sum_{i=1}^{n} h_{UI}$, **BKs** (*Total*) $= \sum_{i=1}^{n} h_i$.

Since E[h_{UA}] = ½×h_A, the ***expected value*** *of* ***RU*** is:

$$E[RU_R]=E[\sum_{i=1}^{n} h_{UI}\,]=\sum_{i=1}^{n} E(h_{UI})=\tfrac{1}{2}\times\sum_{i=1}^{n} h_i \qquad (1).$$

Members Send:

BKs $(A,level\ i) = (h_{UA}\text{-}1)_{UP} + (h_A - i)_{DN(i)} = h_{UA} + h_A\text{-}1\text{-}i$.

$\mathbf{RU_S}(A) = (h_{UA}\text{-}1)_{UP}$ (1 BK/rt) + $(h_{UA}\text{-}1)_{DN}$ (h_A-½h_{UA} BKs/rt).

$\mathbf{RU_S}$(*total*,twice)= $(h_{UA}\text{-}1)$, **BKs**(*total*) $= (h_{UA}\text{-}1+\sum_{i=1}^{h_{UA}-1}(h_A - i))$

$= h_{UA}\times h_A - \tfrac{1}{2}\times h_{UA}\times(h_{UA}\text{-}1) = h_{UA}\times(h_A - \tfrac{1}{2}\times(h_{UA}\text{-}1))$.

Expected Total Sending Cost values for all Members:

$$\boldsymbol{E[RU_S]}=E[\sum_{i=1}^{n} h_{UI}\,]=\sum_{i=1}^{n} E(h_{UI})=\tfrac{1}{2}\sum_{i=1}^{n} h_i \ \text{(used twice)} \qquad (2)$$

$$\boldsymbol{E[BK_S]}=E(\sum_{i=1}^{n} h_{UI}\,(h_I - \tfrac{1}{2}(h_{UI}-1)))=\tfrac{3}{8}\sum_{i=1}^{n} E(h_I^{\,2})+\tfrac{1}{4}\sum_{i=1}^{n} E(h_I) \qquad (3)$$

The ST is not necessarily balanced. Any arbitrary ST can be derived from a balanced one of the same size, since the following observation holds by induction: for a path to be extended from height h to $h+k$, one or more other paths must be abbreviated by k hops in total. The average member participation still remains the same for both phases. Hence:

$$\sum_{i=1}^{n} h_i = \sum_{i=1}^{n} h \ = n\times h \qquad (4),$$

and then (1), (2), (3) can be revisited.

(1), (2), (4) $\Rightarrow E[RU_R]=\tfrac{1}{2}(n\times h),\ E[RU_S]=\tfrac{1}{2}(n\times h)$.

From (3), (4), with the use of probability theory we obtain:

$$E[BK_S]=\tfrac{3}{8}\sum_{i=1}^{n} E(h_I^{\,2})+\tfrac{1}{4}(n\times h)=\tfrac{3}{8}\sum_{i=1}^{n}(E(h_I))^2+\sigma_h^2)+\tfrac{1}{4}(n\times h),$$

where σ_h^2 is the path variance (expected to be low). Hence,

$$E[BK_S]=\tfrac{3}{8}(n\times h^2)+\tfrac{3}{8}(n\times\sigma_h^2)+\tfrac{1}{4}(n\times h).$$

It is clear that DS-TGDH is substantially more efficient w.r.t. the overall communication, and as will be shown next via simulations, it achieves a much better routing performance.

We considered a generic framework to apply and compare the two protocols. Since not all network nodes belong to the secure subgroup, using broadcast and flooding the network is inefficient. The use of multicast brings about several issues: (*a*) not always benefit for network/nodes configuration (i.e. arbitrary relays), (*b*) it should be optimized as well to lower the overhead, incurring substantial complexity, and (*c*) it may not be supported by all heterogeneous network nodes.

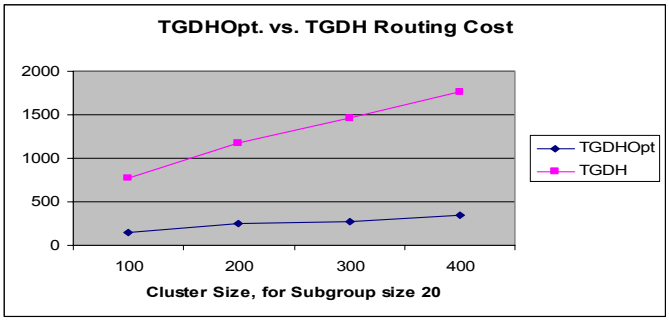
VIII. SIMULATION RESULTS FOR DS-TGDH, TGDH

i) Simulation Set-Up and Discussion:

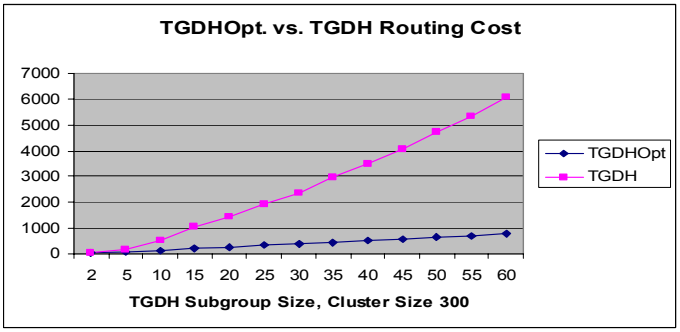
We have conducted simulations to compare the routing cost of DS-TGDH *vs.* TGDH. We use different graphs to generate the secure subgroups in a random manner every time. We use two methods for leader selection: either an arbitrary member or the member with the largest "member" degree becomes the leader. At the end of the subgroup "registration" period, the leader piggybacks the list of members into the routing packets.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

Through routing, each member obtains path(s) to its closest neighbor(s). The concept of "proximity" is dynamically determined by setting a threshold in the hop distance between two members. If no neighbors are found in the proximity, the search diameter (TTL) is gradually expanded. We further assume that while the ST is formed, members' placement and consequently the proximity lists do not change significantly. Such a change could result in a different "optimal" solution, and the one currently generated would become outdated and probably suboptimal. However, our algorithm is fairly fast, so that the topological changes that occur do not "offset" our solution much from the optimal. Even though DS-TGDH is more sensitive to mobility than TGDH, it still reduces significantly the resulting RC.



Graph1: Total routing overhead computation for original TGDH vs. Decentralized TGDH w.r.t. Cluster Size, for Subgroup Size = 20.



Graph2: Total routing overhead computation for original TGDH vs. Decentralized TGDH w.r.t. Subgroup Size, for Cluster Size = 300.

For our evaluation, we generated various random graphs for a given input of the number of nodes n and the number of members m. For the same graph and input, we vary the subgroup configuration, i.e. select randomly the m members. For each graph of input $<n, m>$ and subgroup configuration, we evaluated the total routing cost of DS-TGDH *vs.* TGDH, and we averaged the results for all random graphs with the same inputs $<n, m>$. We have tested the following scenarios: **Cluster Size:** [100,…500], **Subgroup Size**: [2,…60].

ii) Simulation Results:

The following graphs illustrate representative results on the RC produced by DS-TGDH and TGDH, measured in number of hops (relays). KM messaging is very heavy for the network nodes, so our aim is to: (*a*) reduce the total amount of packets required and relieve as many nodes as possible from relaying large key data, and (*b*) dynamically distribute the KM tasks of a single leader to more or potentially all members. Indeed, DS-TGDH results in significant RC savings, and in most cases the associated ratio is: $\frac{DS-TGDH(RC)}{TGDH(RC)} < 0.3$. These savings are even bigger if we consider also the amount of redundant key data, as calculated in our analysis, relayed by nodes during every round of TGDH. To illustrate this with an example, for a cluster of size 300, and a subgroup of size 35, the averaged relays produced are 457 for DS-TGDH, 2987 for TGDH. This means that the total amount of redundant data relayed is: $^{2987}/_{2} \times 34 \times \lceil \log_2 34 \rceil \times 1024$ bits = 311,986,170 bits.

IX. Conclusion

This paper focuses on the design of a decentralized, low cost robust version of TGDH, denoted as DS-TGDH, for a general MANET. DS-TGDH benefits from a combined consideration of the underlying routing and the existing logical KG flow to enhance the performance and behavior of the original ancestor scheme. We provide a topology aware schedule on top of which DS-TGDH is executed, and we analytically evaluate the overhead of the schedule generation and execution, for the initial and the steady state (including membership changes and disruptions). Exploring the potential of DS-TGDH through different views (communication, routing, processing overhead), we attempt to provide accurate and spherical evaluation and understanding of both algorithms, and of their strong and weak assets. Through our analytical work and simulation results we show how we can achieve better performance in the environment of interest, with our scheme.

Acknowledgments

Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. Research also supported by the U.S. Army Research Office under grant No DAAD19-01-1-0494. The U.S. Gover- nment is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon

References

[1] K. Becker, U. Wille, "Communication Complexity of Group Key Distribution," *Proc.5th ACM Conference on Computer & Communicatios Security*, pp. 1-6, San Francisco, CA, November 1998.

[2] M. Steiner, G. Tsudik, M. Waidner, "Diffie-Hellman Key Distribution Extended to Groups," *3rd ACM Conference on Computer & Communication Security*, pp. 31-37 ACM Press, 1996.

[3] M. Hietalahti. "Key Establishment in Ad-Hoc Networks," *M.S. Thesis*, Helsinki University of Technology, Dept. of Computer Science and Engineering, May 2001.

[4] A.Perrig, "Efficient Collaborative Key Management Protocols for Secure Autonomous Group Communication," *Proc. International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC'99),* pp. 192-202, July 1999.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

[5] N.Asokan, P. Ginzboorg, "Key-Agreement in Ad-Hoc Networks," *Computer Communications*, Vol. 23, No. 18, pp. 1627-1637, 2000.
[6] M.Striki, J.Baras, "Scalable and Efficient Key Agreement Protocols for Secure Multicast Communication in MANETs," *Technical Report CSHCN 2002-62*, Institute for Systems Research, University of Maryaland.
[7] D. McGrew, A.T. Sherman, "Key-Establishment in Large Dynamic Groups Using One-Way Function Trees," *IEEE Trans. On Software Engineering*, Vol 29, No. 5, pp. 444-458, 2003.
[8] H. Harney, E.Harder, "Logical Tree Hierarchy Protocol," *Internet Draft*, draft-harney-sparta-lkhp-sec-00.txt, Internet Engineering Task Force, March 1999.
[9] H. Harney, C.Muckenhirn, "Group Key Management Protocol (GKMP) Specification/Architecture," *RFC 2093 and 2094*, , Internet Engineering Task Force, July 1997.
[10] Y. Kim, A. Perrig, G. Tsudik, "Simple and Fault Tolerant Key Agreement for Dynamic Collaborative Groups," *Proc. 7th ACM Conference on Computer and Communication Security (CCS 2000),* pp. 235-244, 2000.
[11] L.Lazos, R.Poovendran, "Cross-Layer Design for Energy Efficient Secure Multicast Communications in Ad Hoc Networks," *Proc. 2004 IEEE International Conference on Communications (ICC '04),* Vol. 6, pp. 3633-3639, Paris, France, June 2004.
[12] Y.Amir, Y.Kim, C.Rotaru, J.Schultz, G.Tsudik, "Exploring Robustness in Group Key Agreement", *Proc. of the 21th IEEE Int'l Conference on Distr. Computing Systems,* pp. 399-408, Phoenix, AZ, April 16-19, 2001.
[13] Y.Amir, Y.Kim, C.Rotaru, J.Schultz, J.Stanton, G.Tsudik, "Secure Group Communication Using Robust Contributory Key Agreement"., *IEEE Trans. on Parallel and Distributed Systems*, Vol. 15, no. 5, pp. 468-480, May 2004.
[14] L.Zhou, Z.Haas, "Securing Adhoc Networks," *IEEE Network Magazine*, vol. 13, no.6, pp. 24-30, Nov/Dec 1999.
[15] J.Kong, P.Zerfos, H.Luo, S.Lu, L.Zhang, "Providing Robust and Ubiquitous Security Support for Wireless Ad-Hoc Networks," *Proc. 2001 IEEE International Conference on Network Protocols (ICNP 2001)*, pp. 251-260, 2001.
[16] S.Capkun, L.Buttyan, J.Hubaux, "Self-Organized Public Key Management for MANET," *IEEE Trans. on Mobile Computing*, Vol. 2, No. 1, pp. 52-64, Jan-Mar. 2003.
[17] L.Eschenauer, V.Gligor., "A Key Management Scheme for Distributed Sensor Networks," *Proc. 9th ACM Conference on Computer and Communication Security (CCS'02)*, pp. 41-47, Nov, 2002.
[18] H.Chan, A.Perrig, D.Song, "Random Key Predistribution Schemes for Sensor Networks," *Proc. 2003 IEEE Symposium on Security and Privacy*, pp. 197-213, May 2003.
[19] M.Striki, J.Baras "A Robust, Distributed TGDH-based Scheme for Secure Group Communications", *Technical Report CSHCN 2005-97*, Institute for Systems Research, University of Maryland, 2005.
[20] A. Perrig, D.Song, D. Trygar, "ELK, a New Protocol for Efficient Large-Group Distribution," *Proc. 2001 IEEE Symposium on Security and Privacy*, pp. 247-262, Oakland, CA, May 2001.
[21] S.Yi, R.Kravets, "Key Management for Heterogeneous Ad hoc Wireless Networks," University of Illinois, Urbana-Champaign, CS dept., TR#UIUCDCS-R-2001-2241, UILU-ENG-2001-1748, July 2002.
[22] S.J.Lee, W.Su, M.Gerla, "Wireless Ad hoc Multicast Routing with Mobility Prediction", *Mobile Networks and Applications*, Vol 6, pp. 351-360, 2001, Kluwer Academic Publishers.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.