Approximation Algorithms for Key Management in Secure Multicast

Agnes Chan[1], Rajmohan Rajaraman[1], Zhifeng Sun[1], and Feng Zhu[2]

[1] Northeastern University, Boston, MA 02115, USA
[2] Cisco Systems, San Jose, CA, USA

Abstract. Many data dissemination and publish-subscribe systems that guarantee the privacy and authenticity of the participants rely on symmetric key cryptography. An important problem in such a system is to maintain the shared group key as the group membership changes. We consider the problem of determining a key hierarchy that minimizes the average communication cost of an update, given update frequencies of the group members and an edge-weighted undirected graph that captures routing costs. We first present a polynomial-time approximation scheme for minimizing the average number of multicast messages needed for an update. We next show that when routing costs are considered, the problem is NP-hard even when the underlying routing network is a tree network or even when every group member has the same update frequency. Our main result is a polynomial time constant-factor approximation algorithm for the general case where the routing network is an arbitrary weighted graph and group members have nonuniform update frequencies.

1 Introduction

A number of data dissemination and publish-subscribe systems, such as interactive gaming, stock data distribution, and video conferencing, need to guarantee the privacy and authenticity of the participants. Many such systems rely on symmetric key cryptography, whereby all legitimate group members share a common key, henceforth referred to as the *group key*, for group communication. An important problem in such a system is to maintain the shared group key as the group membership changes. The main security requirement is *confidentiality*: only valid users should have access to the multicast data. In particular this means that any user should have access to the data only during the time periods that the user is a member of the group.

There have been several proposals for multicast key distribution for the Internet and ad hoc wireless networks [2, 7, 8, 18, 24]. A simple solution proposed in early Internet RFCs is to assign each user a *user key*; when there is a change in the membership, a new group key is selected and separately unicast to each of the users using their respective user keys [8, 7]. A major drawback of such a key management scheme is its prohibitively high update cost in scenarios where member updates are frequent.

The focus of this paper is on a natural key management approach that uses a hierarchy of auxiliary keys to update the shared group key and maintain the desired security properties. Variations of this approach, commonly referred to as the *Key Graph* or the *Logical Key Hierarchy* scheme, were proposed by several independent groups of researchers [2, 4, 21, 23, 24]. The main idea is to have a single group key for data communication, and have a group controller (a special server) distribute auxiliary subgroup keys to the group members according to a key hierarchy. The leaves of the key hierarchy are the group members and every node of the tree (including the leaves) has an associated *auxiliary* key. The key associated with the root is the shared group key. Each member stores auxiliary keys corresponding to all the nodes in the path to the root in the hierarchy. When an update occurs, say at member u, then all the keys along the path from u to the root are rekeyed from the bottom up (that is, new auxiliary keys are selected for every node on the path). If a key at node v is rekeyed, the new key value is multicast to all the members in the subtree rooted at v using the keys associated with the children of v in the hierarchy.[3] It is not hard to see that the above key hierarchy approach, suitably implemented, yields an exponential reduction in the number of multicast messages needed on a member update, as compared to the scheme involving one auxiliary key per user.

The effectiveness of a particular key hierarchy depends on several factors including the organization of the members in the hierarchy, the routing costs in the underlying network that connects the members and the group controller, and the frequency with which individual members join or leave the group. Past research has focused on either the security properties of the key hierarchy scheme [3] or concentrated on minimizing either the total number of auxiliary keys updated or the total number of multicast messages [22], not taking into account the routing costs in the underlying communication network.

1.1 Our contributions

In this paper, we consider the problem of designing key hierarchies that minimize the average update cost, given an arbitrary underlying routing network and given arbitrary update frequencies of the members, which we refer henceforth to as weights. Let S denote the set of all group members. For each member v, we are given a weight w_v representing the update probability at v (e.g., a join/leave action at v). Let G denote an edge-weighted undirected routing network that connects the group members with a group controller r. The cost of any multicast from r to any subset of S is determined by G. The cost of a given key hierarchy is then given by the weighted average, over the members v, of the sum of the costs of the multicasts performed when an update occurs at v. A formal problem definition is given in Section 2.

[3] We emphasize here that auxiliary keys in the key hierarchy are only used for maintaining the group key. Data communication within the group is conducted using the group key.

- We first consider the objective of minimizing the average number of multicast messages needed for an update, which is modeled by a routing tree where the multicast cost to every subset of the group is the same. For uniform multicast costs, we precisely characterize the optimal hierarchy when all the member weights are the same, and present a polynomial-time approximation scheme when member weights are nonuniform. These results appear in Section 3.
- We next show in Section 4 that the problem is NP-hard when multicast costs are nonuniform, even when the underlying routing network is a tree or when the member weights are uniform.
- Our main result is a constant-factor approximation algorithm in the general case of nonuniform member weights and nonuniform multicast costs captured by an arbitrary routing graph. We achieve a 75-approximation in general, and achieve improved constants of approximation for tree networks (11 for nonuniform weights and 4.2 for uniform weights). These results are in Section 5.

Our approximation algorithms are based on a simple divide-and-conquer framework that constructs "balanced" binary hierarchies by partitioning the routing graph using both the member weights and the routing costs. A key ingredient of our result for arbitrary routing graphs is the algorithm of [14] which, given any weighted graph, finds a spanning tree that simultaneously approximates the shortest path tree from a given node and the minimum spanning tree of the graph.

Due to space constraints, we have omitted many of the proofs in this paper. Please refer to the full version of the paper [5] for details.

1.2 Related work

Variants of the key hierarchy scheme studied in this paper were proposed by several independent groups [2, 4, 21, 23, 24]. The particular model we have adopted matches the Key Graph scheme of [24], where they show that a balanced hierarchy achieves an upper bound of $O(\log n)$ on the number of multicast messages needed for any update in a group of n members. In [22], it is shown that $\Theta(\log n)$ messages are necessary for an update in the worst case, for a general class of key distribution schemes. Lower bounds on the amount of communication needed under constraints on the number of keys stored at a user are given in [3]. Information-theoretic bounds on the number of auxiliary keys that need to be updated given member update frequencies are given in [19].

In recent work, [16] and [20] have studied the design of key hierarchy schemes that take into account the underlying routing costs and energy consumption in an ad hoc wireless network. The results of [16, 20], which consist of hardness proofs, heuristics, and simulation results, are closely tied to the wireless network model, relying on the broadcast nature of the medium. In this paper, we present approximation algorithms for a more basic routing cost model given by an undirected weighted graph.

The special case of uniform multicast costs (with nonuniform member weights) bears a strong resemblance to the Huffman encoding problem [11]. Indeed, it can

be easily seen that an optimal *binary* hierarchy in this special case is given by the Huffman code. The truly optimal hierarchy, however, may contain internal nodes of both degree 2 and degree 3, which contribute different costs, respectively, to the leaves. In this sense, the problem seems related to Huffman coding with unequal letter costs [12], for which a PTAS is given in [6]. The optimization problem that arises when multicast costs and member weights are both uniform also appears as a special case of the constrained set selection problem, formulated in the context of website design optimization [10]. Another related problem is broadcast tree scheduling where the goal is to determine a schedule for broadcasting a message from a source node to all the other nodes in a heterogeneous network where different nodes may incur different delays between consecutive message transmissions [13, 17]. Both the Key Hierarchy Problem and the Broadcast Tree problem seek a rooted tree in which the cost for a node may depend on the degrees of the ancestors; however, the optimization objectives are different.

As mentioned in Section 1.1, our approximation algorithm for the general key hierarchy problem uses the elegant algorithm of [14] for finding spanning trees that simultaneously approximates both the minimum spanning tree weight and the shortest path tree weight (from a given root). Such graph structures, commonly referred to as *shallow-light trees* have been extensively studied (e.g., see [1, 15]).

2 Problem definition

An instance of the Key Hierarchy Problem is given by the tuple (S, w, G, c), where S is the set of group members, $w : S \to Z$ is the weight function (capturing the update probabilities), $G = (V, E)$ is the underlying communication network with $V \supseteq S \cup \{r\}$ where r is a distinguished node representing the group controller, and $c : E \to Z$ gives the cost of the edges in G.

Fix an instance (S, w, G, c). We define a *hierarchy* on a set $X \subseteq S$ to be a rooted tree H whose leaves are the elements of X. For a hierarchy T over X, the cost of a member $x \in X$ with respect to T is given by

$$\sum_{\text{ancestor } u \text{ of } x} \; \sum_{\text{child } v \text{ of } u} M(T_v) \tag{1}$$

where T_v is the set of leaves in the subtree of T rooted at v and for any set $Y \subseteq S$, $M(Y)$ is the cost of multicasting from the root r to Y in G. The cost of a hierarchy T over X is then simply the sum of the weighted costs of all the members of X with respect to T. The goal of the Key Hierarchy Problem is to determine a hierarchy of minimum cost.

We introduce some notation that is useful for the remainder of the paper. We use $\mathrm{OPT}(S)$ to denote the cost of an optimal hierarchy for S. We extend the notation W to hierarchies and to sets of members: for any hierarchy T (resp., set X of members), $W(T)$ (resp., $W(X)$) denotes the sum of the weights of the leaves of T (resp., members in X). Our algorithms often combine a set $\mathcal{H}$ of two or three hierarchies to another hierarchy T': `combine`$(\mathcal{H})$ introduces a new root

node R, makes the root of each hierarchy in $\mathcal{H}$ as a child of R, and returns the hierarchy rooted at R.

Using the above notation, a more convenient expression for the cost of a hierarchy T over X is the following reorganization of the summation in Equation 1:

$$\sum_{u \in T} W(T_u) \sum_{\text{child } v \text{ of } u} M(T_v) \tag{2}$$

3 Uniform multicast cost

In this section, we consider the special case of the Key Hierarchy problem where the multicast cost to any subset of group members is the same. Thus, the objective is to minimize the average number of multicast messages sent for an update. We note that the number of multicast messages sent for an update at a member u is simply the sum of the degrees of its ancestors in the hierarchy (as is evident from Equation 1).

3.1 Structure of an optimal hierarchy for uniform member weights

When all the members have the same weight, we can easily characterize an optimal key hierarchy by recursion. Let n be the number of members. When $n = 1$, the key hierarchy is just a single node tree. When $n = 2$, the key hierarchy is a root with two leaves as children. When $n = 3$, the key hierarchy is a root with three leaves as children. When $n > 3$, we are going to build this key hierarchy recursively. First divide n members into 3 balanced groups, i.e. the size of each group is between $\lfloor n/3 \rfloor$ and $\lceil n/3 \rceil$. Then the key hierarchy is a root with 3 children, each of which is the key hierarchy of one of the 3 groups built recursively by this procedure. It is easy to verify that the cost of this hierarchy is given by:

$$f(n) = \begin{cases} 3n\lfloor \log_3 n \rfloor + 4(n-k) \text{ when } k \le n < 2k \\ 3n\lfloor \log_3 n \rfloor + 5n - 6k \text{ when } 2k \le n < 3k \end{cases}$$

The following theorem is due to [9, 10], where this scenario arises as a special case of the constrained set selection problem.

Theorem 1 ([9, 10]). *For uniform multicast costs and member weights, the above key hierarchy is optimal.*

3.2 A polynomial-time approximation scheme for nonuniform member weights

We give a polynomial-time approximation scheme for the Key Hierarchy Problem when the multicast cost to every subset of the group is identical and the members have arbitrary weights. Given a positive constant ε, we present an polynomial-time algorithm that produces a $(1 + O(\varepsilon))$-approximation. We assume that $1/\varepsilon$ is a power of 3; if not, we can replace ε by a smaller constant that satisfies

this condition. We round the weight of every member up to the nearest power of $(1+\varepsilon)$ at the expense of a factor of $(1+\varepsilon)$ in approximation. Thus, in the remainder we assume that every weight is a power of $(1+\varepsilon)$. Our algorithm PTAS(S), which takes as input a set S of members with weights, is as follows.

1. Divide S into two sets, a set H of the $3^{1/\varepsilon^2}$ members with the largest weight and the set $L = S - H$.
2. Initialize $\mathcal{L}$ to be the set of hierarchies consisting of one depth-0 hierarchy for each member of L.
3. Repeat the following step until it can no longer be executed: if T_1, T_2, and T_3 are hierarchies in $\mathcal{L}$ with identical weight, then replace T_1, T_2, and T_3 in $\mathcal{L}$ by `combine`$(\{T_1, T_2, T_3\})$. (Recall the definition of `combine` from Section 2.)
4. Repeat the following step until $\mathcal{L}$ has one hierarchy: replace the two hierarchies T_1, T_2 with least weight by `combine`$(\{T_1, T_2\})$. Let T_L denote the hierarchy in $\mathcal{L}$.
5. Compute an optimal hierarchy T^* for H. Determine a node in T^* that has weight at most $W(S)\varepsilon$ and height at most $1/\varepsilon$. We note that such a node exists since every hierarchy with at least ℓ leaves has a set N of at least $1/\varepsilon$ nodes at depth at most $1/\varepsilon$ with the property that no node in N is an ancestor of another. Set the root of T_L as the child of this node. Return T^*.

We now analyze the above algorithm. At the end of step 3, the cost of any hierarchy T in $\mathcal{L}$ is equal to $\sum_{v \in T} 3w_v \log_3(W(T)/w_v)$. If $\mathcal{L}$ is the hierarchy set at the end of step 3, then the additional cost incurred in step 4 is at most $\sum_{T \in \mathcal{L}} 2W(T) \log_2(W(L)/W(T))$.

Since there are at most two hierarchies in any weight category in $\mathcal{L}$ at the start of step 4, at least $1 - 1/\varepsilon^2$ of the weight in the hierarchy set is concentrated in the heaviest $4/\varepsilon^3$ hierarchies of $\mathcal{L}$. Step 4 is essentially the Huffman coding algorithm and yields an optimal binary hierachy. We can show that this binary hierarchy achieves an approximation of 3. This yields the following bound on the increase in cost due to step 4:

$$3\left(\varepsilon^2 W(L) \log_{1+\varepsilon} 3 + (1-\varepsilon^2) W(L) \log_2(4/\varepsilon^2)\right) \leq W(L)/\varepsilon,$$

for ε sufficiently small. The final step of the algorithm increases the cost by at most $W(L)/\varepsilon + \varepsilon W(S)$. Thus, the total cost of the final hierarchy is at most

$$\begin{aligned} &\mathrm{OPT}(H) + \mathrm{OPT}(L) + W(L)/\varepsilon + W(L)/\varepsilon + \varepsilon W(S) \\ &\leq \mathrm{OPT}(H) + \mathrm{OPT}(L) + 2\varepsilon \mathrm{OPT}(S) + \varepsilon \mathrm{OPT}(S) \\ &\leq (1 + 3\varepsilon)\mathrm{OPT}(S). \end{aligned}$$

(The second step holds since $\mathrm{OPT}(S) \geq \sum_{v \in L} w_v \log_3(W(S)/w_v) \geq W(L)/\varepsilon^2$.)

4 Hardness results

In this section, first we show that Key Hierarchy Problem is strongly NP-complete if group members have nonuniform weights and the underlying routing network is a tree. Then we show the problem is also NP-complete if group members have uniform weights and the underlying routing network is a general graph.

4.1 Weighted key hierarchy problem with routing tree

We refer the reader to [5] for the proof of the following theorem.

Theorem 2. *When group members have different weights and the routing network is a tree, the Key Hierarchy Problem is NP-complete.*

4.2 Unweighted key hierarchy problem

We refer the reader to [5] for the proof of the following theorem.

Theorem 3. *When group members have the same key update weights and the routing network is a general graph, the Key Hierarchy Problem is NP-complete.*

5 Approximation algorithms for nonuniform multicast costs

We first present, in Section 5.1, an 11-approximation algorithm for the case where the underlying communication network is a tree. Then we present, in Section 5.2, a 75-approximation algorithm for the most general case of our problem, where the communication network is an arbitrary weighted graph.

5.1 Approximation algorithms for routing trees

Given any routing tree, let S be the set of members. We start with defining a procedure `partition`() that takes as input the set S and returns a pair (X, v) where X is a subset of S and v is a node in the routing tree. First, we determine if there is an internal node v that has a subset C of children such that the total weight of the members in the subtrees of the routing tree rooted at the nodes in C is between $W(S)/3$ and $2W(S)/3$. If v exists, then we partition S into two parts X, which is the set of members in the subtrees rooted at the nodes in C, and $S \setminus X$. It follows that $W(S)/3 \leq W(X) \leq 2W(S)/3$. If v does not exist, then it is easy to see that there is a single member with weight more than $2W(S)/3$. In this case, we set X to be the singleton set that contains this heavy node which we call v. The procedure `partition`(S) returns the pair (X, v). In the remainder, we let Y denote $S \setminus X$.

ApproxTree(S)

1. If S is a singleton set, then return the trivial hierarchy with a single node.
2. $(X, v) =$ `partition`(S); let Y denote $S \setminus X$.
3. Let Δ be the cost from root to partition node v. If $\Delta \leq M(S)/5$, then let $T_1 =$ApproxTree(X); otherwise $T_1 =$ PTAS(X). (PTAS is the algorithm introduced in Section 3.2.)
4. $T_2 =$ApproxTree(Y).
5. Return `combine`(T_1, T_2).

Theorem 4. *Algorithm* ***ApproxTree*** *is an* $(11+\varepsilon)$*-approximation, where* $\varepsilon > 0$ *can be made arbitrarily small.*

Proof. Let $\mathrm{ALG}(S)$ be the key hierarchy constructed by our algorithm, $\mathrm{OPT}(S)$ be the optimal key hierarchy. In the following proof, we abuse our notation and use $\mathrm{ALG}(\cdot)$ and $\mathrm{OPT}(\cdot)$ to refer to both the key hierarchies and their cost. We notice that $\mathrm{OPT}(S) \geq \mathrm{OPT}(X) + \mathrm{OPT}(Y)$.

We prove by induction on the number of members in S that $\mathrm{ALG}(S) \leq \alpha \cdot \mathrm{OPT}(S) + \beta \cdot W(S)M(S)$, for constants α and β specified later. The induction base case, when $|S| \leq 2$, is trivial. For the induction step, we consider three cases depending on the distance to the partition node v and whether we obtain a balanced partition; we say that a partition (X, Y) is balanced if $\frac{1}{3}W(S) \leq W(X), W(Y) \leq \frac{2}{3}W(S)$. The first case is where $\Delta \leq M(S)/5$ and the partition is balanced. In this case, we have

$$\begin{aligned}
\mathrm{ALG}(S) &= \mathrm{ALG}(X) + \mathrm{ALG}(Y) + W(S)\left[M(X) + M(Y)\right] \\
&\leq \alpha \cdot \mathrm{OPT}(X) + \beta \cdot W(X)M(X) + \alpha \cdot \mathrm{OPT}(Y) + \beta \cdot W(Y)M(Y) \\
&\quad + W(S)\left[M(X) + M(Y)\right] \\
&\leq \alpha\left[\mathrm{OPT}(X) + \mathrm{OPT}(Y)\right] + \left(\frac{2}{3}\beta + 1\right) W(S)\left[M(X) + M(Y)\right] \\
&\leq \alpha \cdot \mathrm{OPT}(S) + \left(\frac{2}{3}\beta + 1\right) W(S)\left[M(S) + \Delta\right] \\
&\leq \alpha \cdot OPT(S) + \beta \cdot w(S)M(S)
\end{aligned}$$

as long as $\left(1 + \frac{1}{5}\right)\left(\frac{2}{3}\beta + 1\right) \leq \beta$, which is true if $\beta \geq 6$. The second case is where $\Delta > M(S)/5$ and the partition is balanced. In this case, we only call the algorithm recursively on Y and use PTAS on X.

$$\begin{aligned}
\mathrm{ALG}(S) &= \mathrm{PTAS}(X) + \mathrm{ALG}(Y) + W(S)\left[M(X) + M(Y)\right] \\
&\leq 5(1+\varepsilon) \cdot \mathrm{OPT}(X) + \alpha \cdot \mathrm{OPT}(Y) + \beta \cdot W(Y)M(Y) \\
&\quad + W(S)\left[M(X) + M(Y)\right] \\
&\leq \alpha \cdot \mathrm{OPT}(S) + \left(\frac{2}{3}\beta + 2\right) W(S)M(S) \\
&\leq \alpha \cdot \mathrm{OPT}(S) + \beta \cdot W(S)M(S)
\end{aligned}$$

as long as $\alpha \geq 5(1+\varepsilon)$ and $\frac{2}{3}\beta + 2 \leq \beta$ which is true if $\beta \geq 6$. The third case is when the partition is not balanced (i.e. $W(X) > \frac{2}{3}W(S)$). In this case, our algorithm connects the heavy node directly to the root of the hierarchy.

$$\begin{aligned}
\mathrm{ALG}(S) &= \mathrm{ALG}(Y) + W(S)\left[M(X) + M(Y)\right] \\
&\leq \alpha \cdot \mathrm{OPT}(Y) + \beta \cdot W(Y)M(Y) + W(S)\left[M(X) + M(Y)\right] \\
&\leq \alpha \cdot \mathrm{OPT}(S) + \frac{1}{3}\beta W(S)M(S) + 2W(S)M(S) \\
&\leq \alpha \cdot \mathrm{OPT}(S) + \beta \cdot W(S)M(S)
\end{aligned}$$

as long as $\frac{1}{3}\beta + 2 \leq \beta$ which is true if $\beta \geq 3$. So, by induction, we have shown $\mathrm{ALG}(S) \leq \alpha \cdot \mathrm{OPT}(S) + \beta \cdot W(S)M(S)$ for $\alpha \geq 5(1+\varepsilon)$ and $\beta \geq 6$. Since $\mathrm{OPT}(S) \geq W(S)M(S)$, we obtain an $(11+\varepsilon)$-approximation. □

If the member weights are uniform, then we can improve the approximation ratio to 4.2 using a more careful analysis of the same algorithm. We refer the reader to [5] for details.

5.2 Approximation algorithms for routing graphs

In this section, we give a constant-factor approximation algorithm for the case where weights are nonuniform and the routing network is an arbitrary graph. In our algorithm, we compute light approximate shortest-path trees (LAST) [14] of subgraphs of the routing graph. An (α, β)-LAST of a given weighted graph G is a spanning tree T of G such that the the shortest path in T from a specified root to any vertex is at most α times the shortest path from the root to the vertex in G, and the total weight of T is at most β times the minimum spanning tree of G.

ApproxGraph(S)

1. If S is a singleton set, return the trivial hierarchy with one node.
2. Compute the complete graph on $S \cup \{root\}$. The weight of an edge (u, v) is the length of shortest path between u and v in the original routing graph.
3. Compute the minimum spanning tree on this complete graph. Call it $\mathrm{MST}(S)$.
4. Compute an (α, β)-LAST L of $\mathrm{MST}(S)$.
5. $(X, v) = \texttt{partition}(L)$.
6. Let Δ be the cost from root to partition node L. If $\Delta \leq M(S)/5$, then let $T_1 = \mathrm{ApproxGraph}(X)$. Otherwise, $T_1 = \mathrm{PTAS}(X)$.
7. $T_2 = \mathrm{ApproxGraph}(Y)$.
8. Return $\texttt{combine}(T_1, T_2)$.

The optimum multicast to a member set is obtained by a minimum Steiner tree, computing which is NP-hard. It is well known that the minimum Steiner tree is 2-approximated by a minimum spanning tree (MST) in the metric space connecting the root to the desired members (the metric being the shortest path cost in the routing graph). So at the cost of a factor 2 in the approximation, we define $M(S)$ to be the cost of the MST connecting the root to S in the complete graph $G(S)$ whose vertex set is $S \cup \{root\}$ and the weight of edge (u, v) is the shortest path distance between u and v in the routing graph.

Theorem 5. *The algorithm **ApproxGraph** is a constant-factor approximation.*

The proof of Theorem 5 is similar to that of Theorem 4. We refer the reader to [5] for the proof details.

References

1. Awerbuch, B., Baratz, A.E., Peleg, D.: Cost-Sensitive Analysis of Communication Protocols. In: PODC (1990)

2. Canetti, R., Garay, J., Itkis, G., Micciancio, D., Naor, M., Pinkas, B.: Multicast Security: A Taxonomy and Some Efficient Constructions. In: INFOCOMM (1999)
3. Canetti, R., Malkin, T., Nissim, K.: Efficient Communication-Storage Tradeoffs for Multicast Encryption. In: EUROCRYPT (1999)
4. Caronni, G., Waldvogel, M., Sun, D., Plattner, B.: Efficient Security for Large and Dynamic Multicast Groups. In: WETICE (1998)
5. Chan, A., Rajaraman, R., Sun, Z., Zhu, F.: Approximation Algorithms for Key Management in Secure Multicast. arXiv:0904.4061v1 [cs.DS] (2009)
6. Golin, M.J., Kenyon, C., Young, N.E.: Huffman coding with unequal letter costs. In: STOC (2002)
7. Harney, H., Muckenhirn, C.: Group Key Management Protocol (GKMP) Architecture. Internet RFC 2094 (1997)
8. Harney, H., Muckenhirn, C.: Group Key Management Protocol (GKMP) Specification. Internet RFC 2093 (1997)
9. Heeringa, B.: Improving Access to Organized Information. Thesis, University of Massachussetts, Amherst (2008)
10. Heeringa, B., Adler, M.: Optimal Website Design with the Constrained Subtree Selection Problem. In: ICALP (2004)
11. Huffman, D.: A Method for the Construction of Minimum-Redundancy Codes. In: IRE (1952)
12. Karp, R.: Minimum-redundancy coding for the discrete noiseless channel. In: IRE Transactions on Information Theory (1961)
13. Khuller, S., Kim, Y.A.: Broadcasting in Heterogeneous Networks. Algorithmica. 14(1), 1–21 (2007)
14. Khuller, S., Raghavachari, B., Young, N.E.: Balancing Minimum Spanning Trees and Shortest-Path Trees. Algorithmica. 14(4), 305–321 (1995)
15. Kortsarz, G., Peleg, D.: Approximating Shallow-Light Trees (Extended Abstract). In: SODA (1997)
16. Lazos, L., Poovendran, R.: Cross-layer design for energy-efficient secure multicast communications in ad hoc networks. In: IEEE Int. Conf. Communications (2004)
17. Liu, P.: Broadcast Scheduling Optimization for Heterogeneous Cluster Systems. J. Algorithms. 42(1), 135–152 (2002)
18. Mittra, S.: Iolus: A Framework for Scalable Secure Multicasting. In: SIGCOMM (1997)
19. Poovendran, R., Baras, J.S.: An information-theoretic approach for design and analysis of rooted-tree-based multicast key management schemes. In: IEEE Transactions on Information Theory (2001)
20. Salido, J., Lazos, L., Poovendran, R.: Energy and bandwidth-efficient key distribution in wireless ad hoc networks: a cross-layer approach. In: IEEE/ACM Trans. Netw. (2007)
21. Shields, C., Garcia-Luna-Aceves, J.J.: KHIP—a scalable protocol for secure multicast routing. In: SIGCOMM (1999)
22. Snoeyink, J., Suri, S., Varghese, G.: A Lower Bound for Multicast Key Distribution. In: IEEE Infocomm (2001)
23. Wallner, D., Harder, E., Agee, R.: Key Management for Multicast: Issues and Architectures. Internet RFC 2627 (1999)
24. Wong, C.K., Gouda, M.G., Lam, S.S.: Secure Group Communications Using Key Graphs. In: SIGCOMM (1998)