Secure Overlay Routing Using Key Pre-Distribution: A Linear Distance Optimization Approach

Mohammed Gharib, *Student Member, IEEE,* Homayoun Yousefi'zadeh, *Senior Member, IEEE,* and Ali Movaghar, *Senior Member, IEEE*

***Abstract*—Key pre-distribution algorithms have recently emerged as efficient alternatives of key management in today's secure communications landscape. Secure routing techniques using key pre-distribution algorithms require special algorithms capable of finding optimal secure overlay paths. To the best of our knowledge, the literature of key pre-distribution systems is still facing a major void in proposing optimal overlay routing algorithms. In the literature work, traditional routing algorithms are typically used twice to find a NETWORK layer path from the source node to the destination and then to find required cryptographic paths. In this paper, we model the problem of secure routing using weighted directed graphs and propose a boolean linear programming (LP) problem to find the optimal path. Albeit the fact that the solutions to boolean LP problems are of much higher complexities, we propose a method for solving our problem in polynomial time. In order to evaluate its performance and security measures, we apply our proposed algorithm to a number of recently proposed symmetric and asymmetric key pre-distribution methods. The results show that our proposed algorithm offers great network performance improvements as well as security enhancements when augmenting baseline techniques.**

***Index Terms*—Overlay Routing, Underlay Routing, Linear Optimization, Shortest Path, Directed Graphs, Key Pre-Distribution.**

I. Introduction

CRYPTOGRAPHY, as a cornerstone of secure communications, requires the use of effective key management algorithms in conjunction with routing algorithms. Trivial secure communications between an arbitrary pair of network nodes requires each node to store $n-1$ pairwise keys in the case of symmetric cryptography and $n-1$ public keys in the case of asymmetric cryptography where n represents the number of network nodes. Storing such large number of keys is not efficient if practical at all in large networks. Key pre-distribution schemes, especially the recently proposed algorithms of [1], [2], [3], [4] all attempt at offering more effective alternatives requiring less resources.

Key pre-distribution schemes suggest storing a small number of keys, say k, at each node from a key pool containing all keys. Such set of keys is named a key ring and is pre-loaded to a node at the initialization phase of the network. Most of the key pre-distribution schemes choose the keys randomly [5], [4], [6] but there are several others that attempt at choosing keys in smarter ways [1], [2], [3], [7]. Since each node stores just a few keys, there may not be a direct secure path among some nodes. In that case, a chain of nodes in which there is a direct path between each adjacent pair form a cryptographic path from the source node to the destination node.

A key pool for key pre-distribution schemes that is built based on symmetric cryptography concepts contains secret pairwise keys. The size of the key pool in addition to the key ring size directly affect cryptographic connectivity of a network. In order to be able to establish a secure communication path, a source node has to find a network layer path to a destination node first. Then, two adjacent nodes forming a network layer hop check whether or not they have at least one key in common. If so, they can communicate securely. Otherwise, they have to find a cryptographic path amongst themselves. In this paper, we refer to the network layer as the underlay layer and the cryptographic layer as the overlay layer. The source node encrypts a message using a pairwise key agreed with its overlay neighbor and sends the message to the overlay neighbor. The neighbor, in turn, decrypts the message and encrypts it with the pairwise key agreed with the next node on the overlay path. This will be repeated till the message reaches the destination node. The number of intermediate decryption-encryption steps is a very important security factor considered for choosing a key pre-distribution method. Trivially, longer overlay paths cause larger numbers of intermediate decryption-encryption steps.

In the case of building key pre-distribution algorithms based on asymmetric cryptography, the key pool has to contain the public keys of all nodes. Thus, the key pool size is equal to the number of nodes, n. In this case, the source node has to first find an overlay path in order to establish secure communication with the destination. The next step is to find an underlay path for each hop within the overlay path. Clearly, the message is decrypted and encrypted just by the intermediate nodes on the overlay path and all other nodes which participate in routing just see the encrypted message.

It is observed that routing using key pre-distribution schemes needs a two layer algorithm capable of finding the underlay path following the corresponding overlay path. In order to find an overlay path to other nodes, each node has to ask other nodes either directly or through intermediary trusted nodes for the IDs of their stored keys implying an already established trust relationship among nodes. One of the inherent problems of this approach is that an adversary node

M. Gharib is with the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran and was a visiting Scholar at the California Institute for Telecommunications and Information Technology, University of California, Irvine. E-mail: gharib@ce.sharif.edu

H. Yousefi'zadeh is with the Center for Pervasive Communications and Computing, University of California Irvine. E-mail: hyousefi@uci.edu

A. Movaghar is with the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. E-mail: movaghar@sharif.edu

may advertise incorrect information in order to insert itself inside a chosen path and then conduct possible attacks.

The main contribution of this paper is proposing a secure routing algorithm jointly optimizing underlay and overlay paths using key pre-distribution schemes but not requiring explicit trust of other network nodes. More specifically, the contributions of this paper are: i) modeling a network using key pre-distribution schemes with directed and weighted graphs, ii) proposing a boolean LP problem for optimal overlay routing in the resulting network graph, iii) analytically reducing the boolean LP problem to a relaxed LP problem and thereby solving the boolean LP in polynomial time, and v) evaluating network performance, security, and energy consumption characteristics of the proposed algorithm for both symmetric and asymmetric key pre-distribution methods operating on top of on-demand routing protocols.

We model a network with a weighted directed graph in which all edges and vertices have their own cost. Then, we propose a secure routing algorithm for the modeled graph using a boolean LP problem. The objective function of our proposed boolean LP problem appears as a linear combination of the overlay and underlay distance. We propose a method to solve this boolean LP problem in polynomial time and show that it is in the general form of a two layer optimal routing for any kind of directed or undirected graph weighted or non-weighted. Hence, our proposed algorithm could be used for secure routing in any network using any key pre-distribution scheme. In order to evaluate the performance and security strength of the proposed algorithm, we apply it to a number of asymmetric and symmetric key pre-distribution schemes proposed in [4], [3], and [2]. Experimental results show that our algorithm improves network performance and enhances network security. The performance metrics evaluated in this work are throughput, delivery completion times, routing control traffic, required storage in each node, and energy consumed for encryption/decryption. The main security metric is the average number of intermediate decryption-encryption steps. As a second security metric, we consider the number of compromised nodes required in order for an attacker to compromise the secrecy of the whole network.

The rest of the paper is organized as follows. Related work is presented in Section II. In Section III, we describe the general problem of overlay routing, introduce our specific formulation of the problem along with the solution to the problem, and the associated analytical work. Experimental results are reported in Section IV. Section V contains the conclusion of our work and a description of its target applications. Finally, Appendix A offers the proof of optimality for the proposed algorithm of Section III.

II. Related Work

Key pre-distribution schemes are categorized into deterministic and probabilistic algorithms. In both categories, each network node is pre-loaded with several keys chosen from a key pool in the initialization phase.

Deterministic key pre-distribution algorithms ensure that there is at least one common key between each pair of neighboring nodes. Assuming the total number of network nodes is n, a trivial deterministic method pre-loads each node with $n-1$ keys. Choi [8], Zhu [9], Çamtepe [1], and Ruj [10] propose different deterministic key pre-distribution schemes. Generally, deterministic key pre-distribution schemes are not scalable and need a rather large storage space. Hence, probabilistic key pre-distribution schemes are preferred over deterministic schemes.

Eschenauer and Gligor in [5] propose the first probabilistic key pre-distribution algorithm in which each pair of neighboring nodes have a common key with a specific probability. The main disadvantage of the basic probabilistic key pre-distribution is that if an attacker compromises several nodes, many links may be potentially rendered insecure. In [6], Chan proposes a more secure algorithm of establishing secure links only between nodes that have at least a given number of common keys. Liu and Ning [11] propose storing bivariate polynomials instead of keys requiring neighboring nodes to have at least one common polynomial. Bechkit [3], Ruj [2], and Çamtepe [1] propose different key pre-distribution schemes based on combinatorial design and the use of symmetric cryptography techniques in which the same keys are used for both encryption and decryption. Other relevant key pre-distribution schemes using asymmetric cryptography techniques in which public-private key pairs are used for encryption and decryption include those proposed in [4], [12], [13].

According to the national institute of standards and technology (NIST), the key length in an asymmetric algorithm has to be at least twice the length of an equivalent strength symmetric algorithm. Hence, many of the proposed key pre-distribution algorithms use symmetric key cryptography. However, the main advantage of asymmetric schemes is that having a compromised node only locally affects the cryptographic neighbors of that node. Moreover, updating a path only requires updating the underlay paths corresponding to cryptographic hops on the overlay path.

In the rest of this section, we describe some of the details of three key pre-distribution schemes used in performance evaluation. Those are symmetric unital key pre-distribution (UKP) [3] and strong Steiner trade (SST) [2] as well as probabilistic asymmetric key pre-distribution (PAKP) [4].

A. Symmetric Key Pre-Distribution Methods

Balanced incomplete block design (BIBD) [14] is a combinatorial design methodology used in key pre-distribution schemes. BIBD arranges v distinct key objects of a key pool into b different blocks each block representing a key ring assigned to a node. Each BIBD design is expressed with a quintuplet (v, b, r, k, λ) where v is the number of keys, b is the number of key rings, r is the number of nodes sharing a key, and k is the number of keys in each key ring. Further, each pair of distinct keys occur together in exactly λ blocks. Any BIBD design can be expressed with the equivalent tuple (v, k, λ) because the relationship $bk = vr$ always holds.

1) Unital Key Pre-Distribution: Unital design is a special case of BIBD design using variable m to represent the design as $(m^3 + 1, m + 1, 1)$. Bechkit [3] proposes a key pre-

distribution method based on unital design suggesting to pre-load each node with t completely disjoint blocks. Referred to as t-UKP method, the method guarantees that the total number of nodes is at most $b/2$ with each distinct pair sharing between zero to t^2 common keys. It is shown that increasing the value of t in t-UKP method leads to increasing the probability of sharing pairwise keys between nodes but decreasing the security strength of the network because each node receives more keys. It is important to note that there is a practical disadvantage in implementing t-UKP method due to the difficulty of designing unitals for large values of m.

2) Strong Steiner Trade: Combinatorial trade or bitrade expressed by $t-(v,k)$ consists of sets $T=\{T_1,T_2\}$ where both T_1 and T_2 contain m blocks of size k chosen from a finite set Π such that the blocks of T_1 and T_2 are completely disjoint. In addition, each set t chosen from Π occurs in exactly the same number of blocks of T_1 as those of T_2. A trade is called Steiner if each set t chosen from Π is repeated at most once in any of the sets T_1 and T_2. Furthermore, such Steiner trade is said to be strong if any block in T_1 and any block in T_2 intersect each other in at most two elements.

Ruj [2] proposes a method to construct SSTs and proves that the proposed construction method results in a $2-(qk,k)$ SSTs where q is a prime number. According to the proposed algorithm, two distinct neighboring nodes can communicate securely if each one of them is from a different set T_1 or T_2 and if they store at least two common keys.

B. Probabilistic Asymmetric Key Pre-Distribution Method

Gharib et al. propose [4] the PAKP method. The method offers a statistical guarantee of cryptographic network connectivity with an average cryptographic path length in the order of $O(\log_k n)$. The authors suggest forming a key pool from the public key of nodes and pre-loading each node with a small number of randomly chosen public keys, say k with $k << n$, instead of pre-loading each node with one public key for each network node. Given the number of network nodes n, they prove that storing $O(\sqrt{n})$ keys results in requiring just one intermediate decryption-encryption step in average while offering a security strength comparable to that of the method of [8].

Table I compares some of the performance and security characteristics of different key pre-distribution methods.

III. Overlay Routing

We open this section by noting that secure message exchange using key pre-distribution forms an overlay layer which can be modeled as a graph. It is important to realize that each hop in an overlay path may consist of several underlay hops. Hence, solving the routing problem requires an algorithm to find the shortest path between a source-destination pair in the overlay network and then find a corresponding underlay path for each overlay hop. Furthermore, the problem for asymmetric key pre-distribution algorithms is more challenging than their symmetric counter parts because secure links between nodes are directional. In symmetric key pre-distribution schemes, each pair of nodes have a bidirectional secure link if they have a pairwise key. Otherwise, there is no direct secure link between them. In this section, we define the secure routing problem of asymmetric schemes and note that they can be applied to symmetric key pre-distribution schemes by making the links bidirectional.

A. Model Description

Secure message exchange by means of probabilistic asymmetric key management introduces an overlay network in which an overlay layer rides on top of the underlay layer. The overlay layer also referred to as the cryptographic layer is formed by the key pre-distribution scheme. It can be modeled with a directed k-regular graph in which each vertex represents a node in the network and each directed edge, e_{ij}, that connects node i to node j represents a stored public key of node j in node i. The underlay routing can be modeled as a unit disk graph (UDG) [15].

The goal is then to find the best path from a source node s to a destination node d. The best path is the path on which both security and performance are optimally measured. The security measure in this context is the number of intermediate decryption-encryption steps that are to be minimized. The performance metric is the shortest underlay path between the source node and the destination node. In this paradigm, security is typically deemed more important than performance. In order to provide a general treatment of the problem, we introduce a tuning parameter, i.e., the vertex cost c_l that controls the weight of security versus performance. In this context, the overall path length is the performance metric while the number of intermediate decryption-encryption steps is the security metric. Choosing a high vertex cost results in a higher cost for longer overlay paths. As a result, the path with a smaller number of decryption-encryption steps, i.e., a shorter overlay path length, is chosen as the optimal path.

Fig. 1 illustrates an example of routing in overlay networks. Light blue links show the underlay connections and directed red links represent the overlay connections. For clarity, not all overlay links are shown in the figure. In order to send a message to its destination node, a source node has to first find an overlay path. There may be several overlay paths from the source node toward the destination. Two such paths are shown in the figure. While the first overlay path reaches the destination through nodes 2 and 3, the second path reaches it through node 5. In the first path, any cryptographic (overlay) hop is formed by just one underlay hop. In the second path, the underlay path for the first cryptographic hop from the source node to node 5, is 2→3→4 where nodes 2, 3, and 4 are underlay neighboring nodes. The red overlay links are directed, i.e., node 1 is able to send a secure message to node 5 directly but not vice versa.

Next, we model the problem with a boolean LP problem and then propose a method to solve this problem in polynomial time, no worse than the time complexity associated with solving the relaxed LP problem without boolean constraints. To model the problem, the network is considered as a k-regular directed graph $G(V,E)$ in which V and E represent sets of vertices and edges, respectively. This graph represents

TABLE I: A comparison of key pre-distribution schemes of interest to this paper.

Scheme	Type	Storage	Link Number	Communication Overhead	Captured Node Resiliency	Intermediate Encryption-Decryption Steps	Mobility Support
2-UKP	Symmetric	$O(k)$	$O(n^2)$	$O(kn)$	$O(\sqrt{n})$	O(Underlay Path Length)	Limited
SST	Symmetric	$O(k)$	$O(n^2)$	$O(kn)$	$O(2.3q)$	O(Underlay Path Length)	Limited
PAKP	Asymmetric	$O(k)$	$O(kn)$	$O(kn)$	$O(n)$	$O(\log_k n)$	Yes

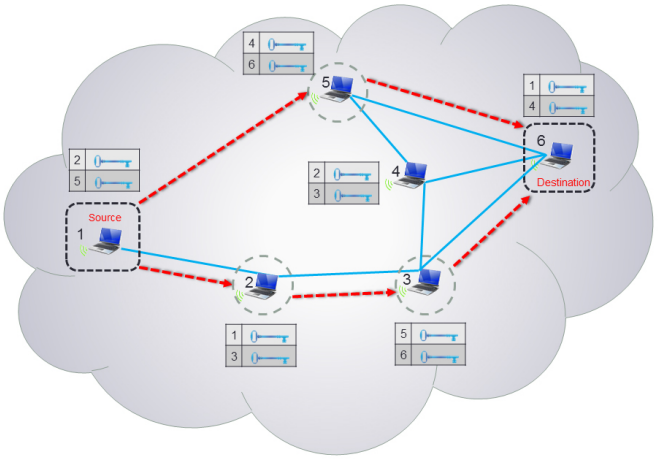


Fig. 1: An example of overlay routing.

an asymmetric key pre-distribution method in which each node stores public keys of exactly k other nodes and has directed links to these nodes. Therefore, each edge $e_{i,j}$ in graph G represents the public key of node j which is stored in node i. Furthermore, in order to model the path length equivalent to each edge in the overlay layer, a cost exactly equal to the shortest underlay path length from node i to node j is assigned for each edge e_{ij} within graph G. Moreover, an additional cost factor associated with intermediate encryption-decryption steps is assigned to each vertex in the graph. Now, we can define the optimal path from the source node s to the destination node d as the path with the overall lowest cost from the source node to the destination node on graph $G(V, E)$.

For an asymmetric key pre-distribution method, we know that the value of k is in the order of $O(\sqrt{n})$ and rather small in comparison with the number of nodes n. Hence, we propose that each node stores a lookup table containing information about stored keys. Moreover, we propose to keep the cost of each edge in the lookup table. The storage needed to store such table is in the order of $O(nk \log n)$ bits. The lookup table contains n rows with each row containing the ID of the public keys stored by its corresponding node. Since $G(V, E)$ is a static graph, all nodes can be pre-loaded with a lookup table in the initialization phase of the network. The pre-loaded table does not contain the cost of edges at this stage. At the network operation phase, each node finds the underlay path length associated with its overlay neighbors by sending simple route requests. In the case of stationary networks, this operation is performed just once at the network initiation phase. In the case of mobile networks, each node has to periodically ask its overlay neighbors for their lookup table update. Note that, since the average of overlay path length is about 2 hops for values of k in the order of $O(\sqrt{n})$, the number of requests is also in the order of $O(\sqrt{n})$. We note that the cost of all vertices is the same representing the cost of an intermediate decryption-encryption step. All together, each node needs to store a lookup table and routing information in addition to the k keys. The storage complexity of storing routing information is in the order of $O(n)$ for proactive protocols and $O(e)$ for on-demand routing protocols where e stands for the number of communicating pairs [16].

B. Problem Formulation

In this section, we formulate an LP problem the answer to which identifies the optimal and most secure path. The problem is to find the path from a source node s to a destination node d with the lowest cost. The proposed LP problem is a boolean problem in which all decision variables assume a value of zero or one. Table II explains the associated notation.

$$\min_{x_{(i,j)}, x_j} \quad \sum_{\substack{(i,j)\in E \\ j\in\{1,\cdots,n\}}} [x_{(i,j)} c_{(i,j)} + x_j c_j] \tag{1}$$

Subject To:

$$\sum_{(s,i)\in E} x_{(s,i)} = 1 \tag{2}$$

$$\sum_{(i,s)\in E} x_{(i,s)} = 0 \tag{3}$$

$$\sum_{(i,d)\in E} x_{(i,d)} = 1 \tag{4}$$

$$\sum_{(d,i)\in E} x_{(d,i)} = 0 \tag{5}$$

$$\sum_{\substack{(i,j)\in E \\ i\neq s \\ j\neq d}} x_{(i,j)} = x_j \tag{6}$$

$$\sum_{\substack{(i,j)\in E \\ i\neq s \\ j\neq d}} x_{(i,j)} = x_i \tag{7}$$

$$x_{(i,j)} \in \{0, 1\} \tag{8}$$

$$x_i \in \{0, 1\} \tag{9}$$

The answer to this LP problem identifies the value of the boolean decision variables $x_{(i,j)}$ for each edge (i, j) and also x_i for each vertex i such that the total path cost is minimized under the given constraints. The value of $x_{(i,j)}$ is one if the

edge (i,j) is chosen as an edge on the optimal path and is set to zero otherwise. Similarly, the value of x_i for each corresponding vertex i is equal to one if the path goes through the vertex or zero otherwise.

TABLE II: The table of notations.

n	Number of network nodes
k	Size of key ring
c_l	Cost of vertex l
(i,j)	Directed link from node i to node j
$c_{(i,j)}$	Cost of link (i,j)
x_l	Boolean value showing whether vertex l is on a path
$x_{(i,j)}$	Boolean value showing whether link (i,j) is on a path
X	$n \times n$ matrix solution of the LP problem
$\Upsilon_{(i,j)}$	Set of vertex disjoint paths from node i to node j
ν_l	Element l within set $\Upsilon_{(i,j)}$
ζ_l	Cost associated with ν_l
γ_l	Decision variable associated with ν_l

The proposed LP problem has six constraints: the first and second constraints guarantee that the source node s is the starting point and does not include any loop. The third and fourth constraints ensure that the destination node d is the end point. The fifth constraint guarantees that if an edge (i,j) is on the optimal path then node j is chosen as a path vertex. The sixth constraint guarantees that if vertex i is chosen on a path, then it has an outgoing edge with a boolean variable equal to one.

Since each outgoing edge should connect to another vertex, the LP problem could be rewritten by removing the vertex variables x_js. The revised problem is described as follows:

$$\min_{x_{(i,j)}} \quad \sum_{\substack{(i,j)\in E \\ j\in\{1,\cdots,n\}}} (c_{(i,j)} + c_j) x_{(i,j)} \tag{10}$$

$$\text{Subject To:} \quad \sum_{(s,i)\in E} x_{(s,i)} = 1 \tag{11}$$

$$\sum_{(i,s)\in E} x_{(i,s)} = 0 \tag{12}$$

$$\sum_{(i,d)\in E} x_{(i,d)} = 1 \tag{13}$$

$$\sum_{(d,i)\in E} x_{(d,i)} = 0 \tag{14}$$

$$\sum_{\substack{(i,j)\in E \\ i\neq s}} x_{(i,j)} = \sum_{\substack{(j,k)\in E \\ j\neq d}} x_{(j,k)} \tag{15}$$

$$x_{(i,j)} \in \{0,1\} \tag{16}$$

where

$$x_{(i,j)} = \begin{cases} 1, & \text{If edge } (i,j) \text{ is on the } \textit{optimal} \text{ path} \\ 0, & \text{Otherwise} \end{cases} \tag{17}$$

The number of variables in the boolean LP problem is $k \times n$, i.e., the number of edges in the k-regular graph. The solution to this problem, for each pair of source and destination nodes, is a boolean vector representing the values of $x_{(i,j)}$. We note that all elements of such vector are equal to zero except for the edges on the optimal path. In the next section, we propose the details of our method for solving the boolean LP problem.

C. Solution Approach

In this section, we propose a method to solve the boolean LP problem of the previous section in polynomial time. It is known that standard LP problems with real-valued decision variables are solvable in polynomial time [17]. A number of techniques may be used to solve LP problems with boolean and integer constraints. Examples include branch and bound [18], branch and cut [19], randomized rounding [20], and the combination of the interior point method with column generating techniques [21]. In some cases, the problem is solved via approximation algorithms [22]. Categorically, all such algorithms either solve the problem with a time complexity much higher than that of solving a relaxed LP problem or cannot guarantee identifying the global optimal solution.

The strength of our algorithm is in solving the boolean LP problem with a time complexity not exceeding that of solving the relaxed LP problem while guaranteeing to identify the optimal solution. We note that our solution approach is generic and can be used to solve a graph optimization problem without any direct relationship with key management or wireless networks.

Our proposed solution is in essence the solution to an LP problem derived by relaxing all of the boolean constraints in the original problem. In Appendix A, we prove that the solution to the relaxed LP problem is the solution to the boolean LP problem.

We start our discussion by noting that the answer to the problem, for each pair of source and destination nodes, could be represented as an $n \times n$ boolean matrix $k \times n$ unknown elements and $(n-k) \times n$ zero elements. Each row in this matrix contains at most one element equal to one and all other elements are zeros. The collection of one elements in the matrix together represent the optimal path.

First, we propose to look at the relaxed LP problem applying the condition $x_{(i,j)} \in [0,1]$ instead of $x_{(i,j)} \in \{0,1\}$. The relaxed problem could be solved in polynomial time using any LP solver such as simplex method [23], interior point method [17], or a recently proposed method based on random walks [24]. The output of the solver will be an $n \times n$ matrix expressing the values of decision variables $x_{(i,j)}$ for each edge $(i,j) \in E$ but it may contain non-integer values. Hence, we just need to start from the row corresponding to the source node, pick the first non-zero element in the row, go to the next row in the matrix according to the column number of the first non-zero element of the current row, and repeatedly take the next step until reaching the destination node. The pseudocode of the proposed algorithm is shown in Algorithm III.1.

As an example, consider the network of Fig. 1 with six nodes and a key ring size equal to two, i.e., $n = 6$ and $k = 2$. In this example, the value of the tuning parameter, i.e., the vertex cost c_l for all vertices l, is set to 2. According to our proposed algorithm, each node at the initialization phase of the network is pre-loaded with two randomly chosen keys and a lookup table. The lookup table of this example is shown in Table III.

Algorithm III.1: OPTIMALPATHFINDER$(s, d, G(V, E))$

comment: s and d are the source and the destination nodes.

$Boolean_{LP} \leftarrow$ BLPDEFINER$(s, d, G(V, E))$;

comment: BLPdefiner function returns the boolean LP problem.

$Relaxed_{LP} \leftarrow$ RELAX$(Boolean_{LP})$;

comment: Relax function returns the relaxed LP problem.

$X \leftarrow$ LPSOLVER$(Relaxed_{LP})$;

comment: LPsolver function returns the solution of the relaxed LP problem as an $n \times n$ matrix X.

$Path \leftarrow s$;

$NextHop \leftarrow s$;

while $d \notin Path$

$NextHop \leftarrow$ COLUMNNUMBER$(X, NextHop)$;

comment: ColumnNumber function returns the column number of the first non-zero element in the corresponding row.

$Path \leftarrow$ CONCATENATE$(Path, NextHop)$;

TABLE III: The lookup table of the source node in Fig. 1.

Node number	1st overlay neighbor	1st neighbor underlay path length	2nd overlay neighbor	2nd neighbor underlay path length
1	2	1	5	4
2	1	1	3	1
3	5	2	6	1
4	2	2	3	1
5	4	1	6	1
6	1	3	4	1

The lookup table of each node is only partially populated at the network initialization phase. At that phase, the contents of columns 3 and 5 of the table above associated with underlay path lengths are not available. The latter implies that a global advance knowledge of the underlay network topology is not required for the operation of our proposed method. However, it is assumed the cryptographic network topology is known. Such assumption is reasonable as cryptographic topology is formed at the time of making key assignments by the key distribution method when the associated information is readily available. Note that key assignment is infrequently changed remaining relatively static even in the presence of mobility. The storage overhead of saving the cryptographic (overlay) topology is in the order of $O(kn)$.

Columns 3 and 5 of the table can be populated in the network initialization phase relying on an information exchange mechanism used between cryptographic (overlay) neighboring pairs in multiple steps. In each step, one-hop overlay neighbors share updates about the underlay path lengths of other nodes as they learn that information. In the first step, nodes learn about the underlay path lengths to their one-hop overlay neighbors. In the second step, nodes learn about the underlay path lengths to their two-hop overlay neighbors and so on. The average number of steps necessary matches the average number of cryptographic hops which is in the order of $O(\log_k n)$ [4]. Underlay network contents of lookup tables can be updated as a function of time or link qualities. In the context of the example of Table III, the elements of underlay path length are populated in row 1 after the first step, rows 2 and 5 after the second step, and rows 3, 4, and 6 after the third step.

Having explained the process of loading and populating the lookup tables, we can now focus on finding the optimal path from source node 1 to destination node 6. Considering a vertex cost of $c_l = 2$, the output of the relaxed LP problem solved with the interior point method is the matrix shown in Equation (18).

$$X = \begin{pmatrix} 0 & 0.45 & 0 & 0 & 0.55 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{18}$$

According to our proposed algorithm and starting from the source node, we begin by checking the first row. The first non-zero element in the first row is the second element. Thus, we consider node number 2 as the first node on the path and go to the second row. In the second row, the first non-zero element is the third element. Hence, the second hop points to node number 3. Completing the steps, the extracted path will be 1→2→3→6 starting from the source node and ending at the destination node.

In Appendix A, we prove all paths extracted from matrix X are optimal and the path identified by Algorithm III.1 is one of those optimal paths. In the context of current example, the latter means the other path extracted from matrix X, i.e., 1→5→6 is also optimal. Here, the cost of the optimal path is equal to 7.

IV. SIMULATION AND EXPERIMENTAL RESULTS

In this section, the performance of our proposed algorithm is evaluated for a number of key pre-distribution algorithms.

A. Simulation Settings

In order to evaluate the performance of our proposed algorithm, we apply it to three key pre-distribution methods, namely, 2-UKP, SST, and PAKP running on top of ad-hoc on-demand distance vector (AODV) routing protocol. We note that our intent is not to compare different key pre-distribution methods against each other, but rather to illustrate how our proposed algorithm can be agnostically applied to solve the secure overlay routing problem of a variety of choices of routing protocols and key pre-distribution methods.

Network simulator NS2 [25] running on Linux is used for our simulations. We choose the interior point method [17] as our LP solver due to its practical efficiency.

A random network for a number of nodes, starting from 100 and increasing to 200 nodes in increments of 10 is

simulated in a 300×300 square meter area. In order to be able to evaluate key pre-distribution algorithms, we generate a fully connected underlay network. The nodes are assumed to be mobile following the random walk mobility model of NS2. The range of movements within the random walk mobility model is set with varying speeds in the interval $[0 \quad 5]$ meters per second and zero pause times. Further, the distance model with a communication range of 100 meters is chosen for message exchange among nodes. We note that the choice of distance model is not taking away from the practicality of evaluation as it can be replaced with a more sophisticated wireless channel model proposed in [26]. The channel bandwidth is set to $1Mbps$. While not limited to AODV, all simulations are performed using AODV routing protocol. Different scenarios are simulated with a different number of connections between 10 and 20. All connections are chosen randomly but once selected the same connections are used for comparing different algorithms in order to keep the comparisons fair. The generated traffic is FTP running on TCP Tahoe. For each connection, the source node sends a file with an average size of $5MB$ to its destination. All simulations are repeated 20 times and the results represent the average values calculated over all runs.

For the key pre-distribution phase in the 2-UKP scenario, the unital design of [27] with unitals of order $m = 4$ is used to simulate a 100 node network. For larger networks, the design of [28] is used to generate unitals of order $m = 5$. The order of unitals is chosen as the minimum value required to generate key rings equal to the number of nodes. Since the key ring size is directly related to the order of selected unitals, the key ring size is set to 10 for networks with 100 nodes and 12 for larger networks.

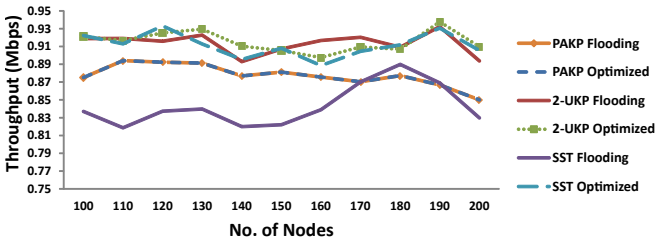
To keep the simulation scenarios fair, the key ring size in SST method is set to $k = 10$. We set $q = 11$ as the best prime number satisfying the inequality $q > k$. Clearly, the maximum number of possible key rings with the chosen parameters is $2q^2 = 242$.

For the PAKP method, we set $k = 10$ to satisfy the relationship $k = O(\sqrt{n})$ and also measure up fairly in comparison to other methods. The value of the vertex cost in PAKP method is set to $n - 1$ in order to guarantee a pair of cryptographic hops is never chosen instead of a single cryptographic hop for the purpose of improving performance.

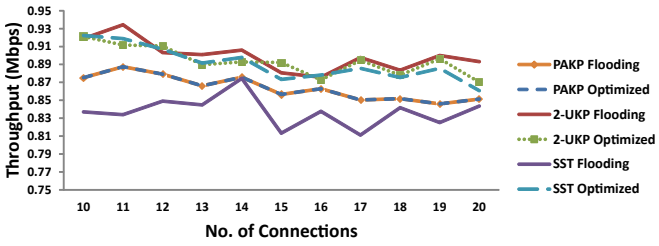
To use the proposed optimal routing method for 2-UKP and SST, we use bidirectional edges between each pair of nodes that share a pairwise key. We note that both methods first identify the underlay path and then for each hop on the underlay path choose the shortest overlay path. Accordingly, the cost of each underlay link is calculated and stored in the lookup table of each node. The implementation of the algorithm exactly follows what was explained in the previous section.

B. Throughput and Latency Comparisons

In this subsection, the measurement results of network throughput associated with successful packet delivery and latency associated with average FTP completion times are reported.



(a)



(b)

Fig. 2: A comparison of average network throughput under random walk mobility model for different key pre-distribution methods as a function of (a) number of nodes and (b) number of connections.

Fig. 2 represents a throughput comparison of different scenarios. Fig. 2a expresses throughput for 10 fixed connections and a different number of nodes. It is observed that the most effective factor impacting throughput is the underlay path chosen in different scenarios. For both SST and 2-UKP methods, the proposed routing algorithm represents a reasonable throughput improvement, especially, in the case of SST in which there is a lower probability of sharing a pairwise key between two adjacent nodes. Since the symmetric key pre-distribution methods find the underlay path and then the shortest overlay path for each hop on the underlay path, the total path length cannot be optimal.

In the case of PAKP method, there is no considerable improvement as the result of applying our proposed routing algorithm. This is alluded to the fact that routing is based on the shortest overlay path from the source node to the destination and the high vertex cost compared to a underlay hop cost. Furthermore and since the underlay network is chosen such that all nodes have a path to each other, increasing the number of nodes leads to increasing the number of neighbors for each node in turn resulting in improving network throughput. However, the improvement is not significant since the network is not very large. While not reported here, we have observed that the throughput improvements in the case of large networks are much more significant.

Fig. 2b represents throughput measures for a different number of connections starting from 10 and increasing to 20 in a 100 node setting. Since increasing the number of connections leads to increasing congestion, the network throughput is slightly decreased as the number of connections increases. Nonetheless, the figure clearly illustrates the improvement in network throughput as the result of applying our algorithm.

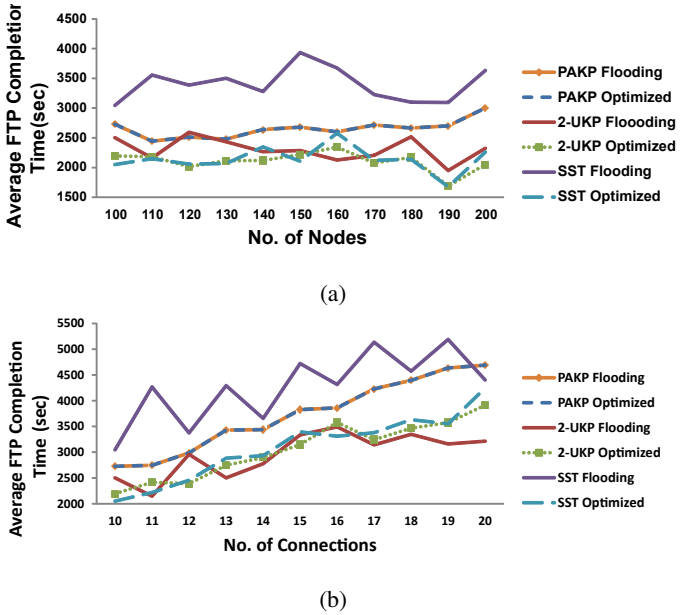


Fig. 3: A random walk mobility comparison of average FTP completion time using different key pre-distribution methods as a function of (a) number of nodes and (b) number of connections.

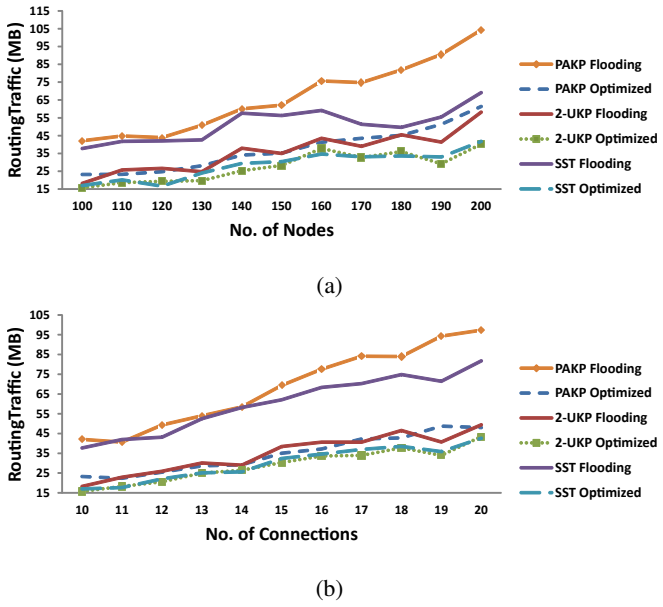


Fig. 4: A random walk mobility comparison of average routing traffic volume for different key pre-distribution methods as a function of (a) number of nodes and (b) number of connections.

Next, average FTP completion times are reported as the metric of latency. Each connection starts sending a 5MB file at a time randomly chosen within the interval $[0, 60]$ seconds. The FTP completion time is calculated as the time interval starting from the transmission time of the first packet and ending at the receipt of the complete file at the receiver. The average FTP completion time is illustrated in Fig. 3 for a different number of nodes and a different number of connections. The FTP completion time improvement for the proposed algorithm in SST and 2-UKP is very clear. Since the underlay path length is the most effective factor on FTP completion time, there is no significant difference between the optimized and non-optimized PAKP results, similar to the case of throughput. Fig. 3a and Fig. 3b present the improvements in FTP completion time as the result of increasing the number of nodes and the number of connections.

C. Traffic and Storage Overhead Comparisons

In this subsection, we first report the results of measuring the routing traffic generated for sending encrypted files of 5MB size. In symmetric key pre-distribution algorithms, each node has to send the IDs of all keys stored in its key ring inside the routing packets. Accordingly, the size of routing packets is increased. In contrast, PAKP does not need to send any extra information in its routing packets. Nonetheless, each node needs to trust other nodes with which it communicates in standard key pre-distribution schemes of interest. Implementing the proposed algorithm eliminates the need for trusting other nodes in addition to decreasing the routing traffic. Fig. 4 represents the generated routing traffic size measured in Megabytes. The improvements in the volume of routing traffic as the result of applying our proposed algorithm is very clear for all schemes. As expected, increasing the number of nodes or the number of connections results in increasing routing traffic in the network.

Next, we discuss the storage overhead on a per node basis. The storage required for different key pre-distribution schemes is also measured as a performance metric. Table IV represents a comparison of the average required storage in each node for different schemes. Table data are derived for a network with 100 nodes and 10 communicating pairs where each node stores 10 keys. Each symmetric key length is considered to be 80 bits. In order to achieve an equivalent strength asymmetric algorithm, an asymmetric key length of 160 bits using elliptic curve cryptography (ECC) is utilized.

D. Energy Consumption and Security Strength Comparisons

In this subsection, we provide experimental results associated with energy consumption and security strength of different methods.

First, we compare the energy consumption associated with performing encryption and decryption using different key pre-distribution schemes before and after applying our proposed algorithm. In order to compensate against the faster speed of symmetric cryptography in comparison to asymmetric cryptography, we force each pair of nodes to agree on a pairwise key for encryption and decryption in the PAKP method. The key agreement process is done using elliptic curve cryptography using Diffie-Hellman method [29]. Fig. 5 represents a comparison of the average energy cost of encryption and decryption associated with different methods before and after applying our proposed algorithm. The energy consumption is calculated according to [30].

TABLE IV: A storage comparison of different key pre-distribution schemes.

Scheme	Number of Links	Lookup Table Size (Kb)	Key Size (b)	Key Ring Size (b)	Average Routing Information Stored in a Node (b)	Total Storage Required in Each Node (Kb)
2-UKP	8404	65.66	80	800	20.92	66.46
SST	1588	12.41	80	800	22.68	13.21
PAKP	1000	7.81	160	1600	10	9.38

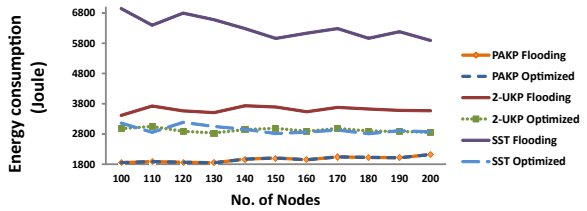


Fig. 5: A comparison of energy cost associated with encryption and decryption of different methods before and after applying our proposed overlay routing algorithm.

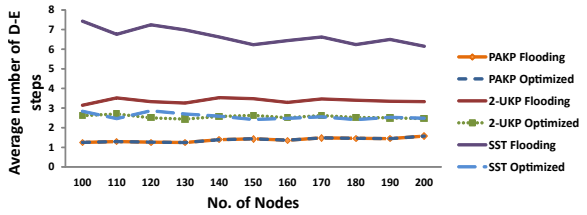


Fig. 6: The average number of decryption-encryption steps for a different number of nodes.

Fig. 5 shows significant improvements in energy consumption of SST and 2-UKP schemes after applying our proposed algorithm. In the case of PAKP scheme, there is no significant improvement to energy consumption since the overlay path length almost remains the same before and after applying our proposed algorithm.

Next, we compare the security strengths of different methods. While network security may be affected by many factors, two factors are of special importance. The first factor is the number of intermediate decryption-encryption steps. In all key pre-distribution methods, messages may have to be decrypted and encrypted several times on their route from source to destination nodes since there may be no direct secure path between some pairs of nodes. A higher number of intermediate decryption-encryption steps increases the probability of an adversary node accessing messages. Fig. 6 shows the average number of decryption-encryption steps associated with different methods.

As mentioned earlier, the proposed algorithm allows for addressing the trade off between performance and security. Furthermore, the average number of intermediate decryption-encryption steps is the same for our proposed algorithm and the algorithm of [4] considering the applied vertex cost. In other methods, especially in SST, there is a significant improvement in the average number of decryption-encryption steps after applying our proposed algorithm.

The second factor that directly affects the security of a key pre-distribution methods is the number of compromised nodes an attacker needs in order to compromise the security of the network as a whole. Since distributed keys are public in PAKP, an attacker can only retrieve the private key of a compromised node itself and no other security information will be lost. Therefore, an attacker needs to compromise all nodes in a PAKP network in order to compromise the whole network. In symmetric key pre-distribution methods, compromising a single node may affect the security of many other links not directly related to the node since the keys distributed between nodes are secret. In 2-UKP method, an attacker needs to compromise just $O(m^2) = O(\sqrt{n})$ nodes in order to retrieve all keys of the key pool. In SST method, an attacker needs to compromise $O(\eta q)$ nodes with $\eta = 2.3$ in order to retrieve all keys of the key pool. The number of compromised nodes leading to compromising the whole network in our simulated scenarios are reported below. Our results show that an attacker needs just 13 nodes to compromise a network implementing 2-UKP key pre-distribution with unitals of order $m = 4$. This number is 24 for unitals of order $m = 5$. For SST key pre-distribution scheme, this number is 25 associated with the choice of $q = 11$. For PAKP, this number is always $n - 1$, i.e., 199 in the case of 200-node simulation scenario.

We close this section by commenting on specific effects of mobility and the scalability of our proposed algorithm. First, we note that the mobility results of Fig. 2, 3, and 4 represent similar patterns as those of stationary networks not reported here albeit the fact that traffic overheads are about 10% higher and delays are about 10% to 20% longer in the cases of mobile scenarios of our experiments. Second, we note that our algorithm scales well in practice and performs robustly with respect to the changes of factors such as node numbers, data loads, and transmission ranges. While not shown here, our experiments with 500 nodes covering a variety of loads and transmission ranges have revealed no significant impact to the pattern of reported results.

V. Conclusion

In this paper, we proposed an optimal secure overlay routing method using key pre-distribution and solved the associated boolean LP problem in polynomial time. We noted the main advantage of our algorithm as being able to solve the optimal routing problem for any graph either directed or undirected as well as weighted or unweighted. We discussed the space complexity of the proposed algorithm to be the storage needed for a lookup table with a size in the order of $O(nk \log n)$ bits. Using network simulator NS2, the proposed algorithm was simulated for a number of different symmetric

and asymmetric key pre-distribution algorithms running on top of AODV routing protocol. The results showed that the use of the proposed algorithm has a significant impact on improving a variety of performance metrics, i.e., decreasing average FTP completion time, reducing routing control traffic, increasing throughput, and improving energy consumption of decryption and encryption. The results also showed that the use of our algorithm enhances the security strength of the network by decreasing the number of intermediate decryption-encryption steps thereby significantly reducing the need for trusting additional nodes.

We conclude this section by discussing the applications of interest to our work. We view our work as a more practical alternative for use in secure network routing applications requiring key distribution. Examples of such applications include pre-planned MANETs used in mission critical and emergency response networks. In such applications, public key infrastructure (PKI) or identity-based cryptography (IBC) methods are used as the key distribution methods. Both PKI and IBC methods need infrastructure and also central servers having knowledge of the complete set of keys and their assignments. Such applications are also typically involved with routing in multiple encrypted and decrypted domains and are subject to high overheads associated with IP address mappings between routing domains with isolated address spaces. Our proposed work introduces a low overhead alternative eliminating the need for infrastructure and central servers as well as the need for multiple routing domains at the cost of storing a small number of per node keys and negligible additional cost of encryption-decryption.

APPENDIX A
PROOF OF OPTIMALITY OF ALGORITHM III.1

In this appendix, we prove that the answer of the relaxed LP problem identified by Algorithm III.1 is in fact the answer to the original boolean LP problem of Section III-B.

Assumption A.1: For any source-destination pair, there exists at least one path from the source node to the destination.

Under the assumption above, there exists at least one optimal path from a source node to its destination. We prove that our proposed algorithm will find this path. We further note that there may exist more than one path with the same optimal objective function value and therefore the optimal path may not be unique.

Let set P be the set of all optimal paths containing at least one path. The boolean LP problem returns an optimal path $p_i \in P$ as its solution. Now, we prove the following lemmas.

Lemma A.2: The output of the relaxed boolean LP problem solved using any LP solver forms a directed graph $G_{out}(V_{out}, E_{out})$. The directed graph contains just one starting vertex, i.e., the source node and just one terminal vertex, i.e., the destination node in addition to a set of intermediate nodes. Moreover, the directed graph does not contain any loop.

Proof: Constraint (11) guarantees that there exists an outgoing non-zero edge from the source node. Constraint (13) guarantees that there exists at least one non-zero incoming edge to the destination. Constraint (15) guarantees that intermediate nodes can be neither the starting nor the terminal vertex because if they have a non-zero incoming edge, then they should have an outgoing edge. Moreover, since graph G_{out} is the optimal solution for the problem and any loop may introduce extra cost, there exists no loop within the graph. ■

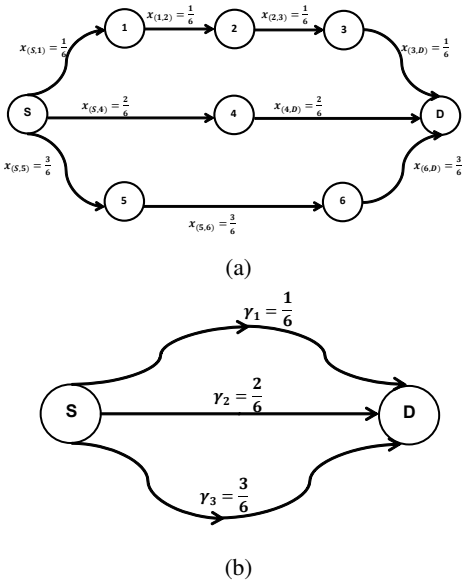


Fig. 7: A sample solution of the relaxed LP problem derived by an LP solver contains just vertex disjoint paths: (a) the output graph, and (b) the equivalent single edge VDP representation.

definition A.3: A set of vertex disjoint paths (VDPs) is defined as the set of paths with a common starting vertex i and a common ending vertex j without any other common intermediate vertices. We refer to such a set as $\Upsilon_{(i,j)}$. We represent element l within this set as ν_l and the associated cost and decision variable as ζ_l and γ_l, respectively.

Lemma A.4: If graph G_{out} contains a number of VDPs, say m, then the value of all link decision variables $x_{(i,j)}$ on each VDP is the same.

$$\{x_{(i,j)}|(i,j) \in \nu_l\} = \{x_{(j,k)}|(j,k) \in \nu_l\}, \quad \forall \nu_l \in \Upsilon_{(i,j)},\ l \in \{1,2,...,m\},\ i,j,k \in \{1,2,...,n\} \tag{19}$$

Therefore, each VDP can be replaced with a single equivalent edge containing a link decision variable equal to one of its link decision variables. Furthermore, the cost of the equivalent edge is equal to the sum of the costs of all edges and vertices on the path.

$$\zeta_l = \sum_{(i,j)\in\nu_l} (c_{(i,j)} + c_j), \quad l \in \{1,2,..,m\} \tag{20}$$

Proof: According to Constraint (15), the sum of the link decision variables of the incoming edges is equal to the sum of the link decision variables of all outgoing edges for each intermediate vertex. In the case of VDPs, each node has just one incoming and one outgoing non-zero edge. Therefore, the value of the link decision variable of one should be equal to that of the other. Thus and as shown by the sample example of 7, we can substitute each VDP with an equivalent edge. ■

Lemma A.5: If the output graph contains just VDPs, then the total cost of each path is equal to the total cost of any other path on the output graph.

$$\zeta_1 = \zeta_2 = ... = \zeta_m \tag{21}$$

Proof: Assume the output graph G_{out} contains m VDPs. According to Lemma A.4, substitute all VDPs with their equivalent single edge representations and further calculate all costs and variables for the new graph. Due to Constraint (11), $\sum_{l=1}^{m} \gamma_l = 1$. Assume the cost of ν_l is more than that of other terms. In this case, removing ν_l and adding the value of its link decision variable to one of other paths, reduces the total value of the objective function. Therefore, the output graph G_{out} has no optimal value. As such, if the output graph generated with the LP solver contains several vertex disjoint paths, then the costs of all paths are equal. ■

Lemma A.6: If the output graph G_{out} contains several paths with just one common intermediate node, then the values of decision variables on all edges on each sub-path are equal. Furthermore, the total costs of all sub-paths from the common to the destination node are equal. Likewise, the costs of all sub-paths from the source to the common node are equal.

Proof: Just like the proof of Lemma A.4, the values of decision variables on all edges of each sub-path are equal. Hence, we can substitute each sub-path with an equivalent edge as transitioned from Fig. 8a to Fig. 8b. Due to Constraint (15), the sum of all decision variables on incoming edges is equal to the sum of decision variables on all outgoing edges where that summation is lower than or equal to 1. Suppose there are m outgoing edges from the common to the destination node. Hence, all outgoing sub-paths are of equal cost following Lemma A.5. Since all sub-paths from the common to the destination node have equal costs, all sub-paths from the source to the common node should have an equal cost to keep the solution optimal. ■

Lemma A.7: If the output graph contains several paths with just one common intermediate node, then exactly p equal cost paths exist where p is the product of the number of source to the common intermediate node paths and the number of intermediate node to destination paths.

Proof: According to Lemma A.6, the output graph has several equal cost sub-paths, i.e., v, to the intermediate common node. There are also several equal cost sub-paths from the intermediate node to the destination, i.e., w. Clearly, any combination of v sub-paths and w sub-paths can form a path from the source node to the destination. As illustrated by Fig. 8c, the number of equal cost paths from the source to the destination is hence equal to $p = v \times w$. ■

Theorem A.8: If the output graph $G_{out}(V_{out}, E_{out})$ contains several common intermediate nodes, the cost of any path on G_{out} is equal to others. Thus, the graph G_{out} can be represented with an equivalent graph containing several VDPs in which all of those VDPs have the same cost.

Proof: According to Lemma A.5, all vertex disjoint sub-paths from a common node to a destination node can be substituted with equal cost equivalent edges. Furthermore, we can replace the common node with several VDPs according to Lemma A.7. Hence, we can start from a common intermediate node placed close to the destination node, remove all common nodes one by one and end up with a new graph containing just several equivalent cost VDPs. The graph G_{out} can then be represented with an equivalent graph containing several equivalent cost vertex disjoint paths. ■

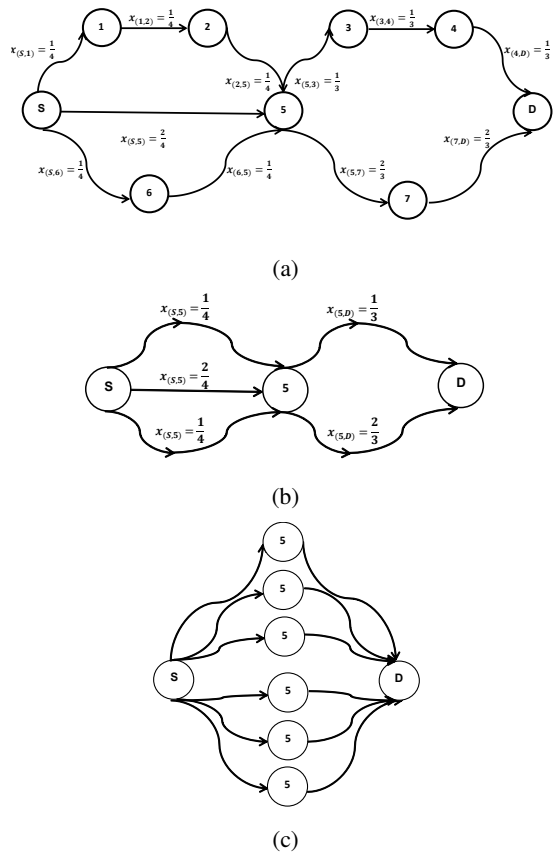


Fig. 8: A sample output graph containing just one intermediate common node: (a) output graph, (b) equivalent form containing the common node, and (c) equivalent vertex disjoint paths.

Corollary A.9: If the boolean LP problem defined in Section III-B is relaxed and solved using an LP solver, any output path solution is optimal. Hence, any directed graph $G(V, E)$ can be modeled with the boolean LP problem defined in Section III-B and solved in polynomial time. Since each undirected graph could be represented as a directed graph in which all edges are bidirectional, the problem of finding such optimal path in any graph is solvable in polynomial time.

Proof: According to Theorem A.8, the output graph can be shown in an equivalent form that contains several equal cost paths. It is also clear that no other path of lower cost exists because such path makes the output graph non-optimal. Hence, any path within the output graph resulted from solving the relaxed boolean LP problem is optimal. ■

We conclude from Corollary A.9 that if the optimal path between the source node and the destination is unique, the output graph will contain just one path from the source to the destination while the decision variables $x_{(i,j)}$ of all edges within the path assume a value of one. Otherwise, the output graph may have several equal cost paths.

REFERENCES

[1] S. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," *Networking, IEEE/ACM Transactions on*, vol. 15, no. 2, pp. 346–358, April 2007.

[2] S. Ruj, A. Nayak, and I. Stojmenovic, "Fully secure pairwise and triple key distribution in wireless sensor networks using combinatorial designs," in *INFOCOM, 2011 Proceedings IEEE*, April 2011, pp. 326–330.

[3] W. Bechkit, Y. Challal, A. Bouabdallah, and V. Tarokh, "A highly scalable key pre-distribution scheme for wireless sensor networks," *Wireless Communications, IEEE Transactions on*, vol. 12, no. 2, pp. 948–959, February 2013.

[4] M. Gharib, E. Emamjomeh-Zadeh, A. Norouzi-Fard, and A. Movaghar, "A novel probabilistic key management algorithm for large-scale manets," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, March 2013, pp. 349–356.

[5] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, ser. CCS '02. New York, NY, USA: ACM, 2002, pp. 41–47. [Online]. Available: http://doi.acm.org/10.1145/586110.586117

[6] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, May 2003, pp. 197–213.

[7] M. Gharib, M. Minaei, M. Golkari, and A. Movaghar, "Expert key selection impact on the manets' performance using probabilistic key management algorithm," in *Proceedings of the 6th International Conference on Security of Information and Networks*, ser. SIN '13. New York, NY, USA: ACM, 2013, pp. 347–351. [Online]. Available: http://doi.acm.org/10.1145/2523514.2523556

[8] T. Choi, H. B. Acharya, and M. Gouda, "The best keying protocol for sensor networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, June 2011, pp. 1–6.

[9] S. Zhu, S. Setia, and S. Jajodia, "Leap: Efficient security mechanisms for large-scale distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ser. CCS '03. New York, NY, USA: ACM, 2003, pp. 62–72. [Online]. Available: http://doi.acm.org/10.1145/948109.948120

[10] S. Ruj and B. Roy, "Key predistribution using combinatorial designs for grid-group deployment scheme in wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 6, no. 1, pp. 4:1–4:28, Jan. 2010. [Online]. Available: http://doi.acm.org/10.1145/1653760.1653764

[11] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ser. CCS '03. New York, NY, USA: ACM, 2003, pp. 52–61. [Online]. Available: http://doi.acm.org/10.1145/948109.948119

[12] A. Liu and P. Ning, "Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, April 2008, pp. 245–256.

[13] Z. Liu, J. Ma, Q. Huang, and S. Moon, "Asymmetric key pre-distribution scheme for sensor networks," *Wireless Communications, IEEE Transactions on*, vol. 8, no. 3, pp. 1366–1372, March 2009.

[14] E. F. Assmus and J. D. Key, *Designs and their codes*. Cambridge University Press, 1992.

[15] M. Huson and A. Sen, "Broadcast scheduling algorithms for radio networks," in *Military Communications Conference, 1995. MILCOM '95, Conference Record, IEEE*, vol. 2, Nov 1995, pp. 647–651 vol.2.

[16] X. Hong, K. Xu, and M. Gerla, "Scalable routing protocols for mobile ad hoc networks," *Network, IEEE*, vol. 16, no. 4, pp. 11–21, Jul 2002.

[17] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, Dec. 1984. [Online]. Available: http://dx.doi.org/10.1007/BF02579150

[18] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Springer, 2000.

[19] M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Rev.*, vol. 33, no. 1, pp. 60–100, Feb. 1991. [Online]. Available: http://dx.doi.org/10.1137/1033004

[20] P. Raghavan and C. D. Thompson, "Randomized rounding: a technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.

[21] A. Vannelli, "An adaptation of the interior point method for solving the global routing problem," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 10, no. 2, pp. 193–203, Feb 1991.

[22] V. V. Vazirani, *Approximation Algorithms*. Springer, 2001.

[23] K. G. Murty, *Linear programming*. John Wiley & Sons, 1983.

[24] D. Bertsimas and S. Vempala, "Solving convex programs by random walks," *J. ACM*, vol. 51, no. 4, pp. 540–556, Jul. 2004. [Online]. Available: http://doi.acm.org/10.1145/1008731.1008733

[25] (2014). [Online]. Available: http://www.isi.edu/nsnam/ns/

[26] H. Yousefi'zadeh and H. Jafarkhani, "An optimal power-throughput tradeoff study for mimo fading ad-hoc networks," *Communications and Networks, Journal of*, vol. 12, no. 4, pp. 334–345, Aug 2010.

[27] S. D. Stoichev and V. D. Tonchev, "Unital designs in planes of order 16," *Discrete Applied Mathematics*, vol. 102, no. 12, pp. 151 – 158, 2000.

[28] (2014) Experimental results of the search for unitals in projective planes of order 25. [Online]. Available: http://sdstoichev1.wordpress.com/2012/02/16/experimental-results-of-the-search-for-unitals-in-projective-planes-of-order-25/

[29] "Ieee standard specifications for public-key cryptography," *IEEE Std 1363-2000*, pp. 1–228, Aug 2000.

[30] N. Potlapally, S. Ravi, A. Raghunathan, and N. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 2, pp. 128–143, Feb 2006.

[31] Y. Zhou, Y. Fang, and Y. Zhang, "Securing wireless sensor networks: a survey," *Communications Surveys Tutorials, IEEE*, vol. 10, no. 3, pp. 6–28, Third 2008.

[32] J. V. D. Merwe, D. Dawoud, and S. McDonald, "A survey on peer-to-peer key management for mobile ad hoc networks," *ACM Comput. Surv.*, vol. 39, no. 1, Apr. 2007. [Online]. Available: http://doi.acm.org/10.1145/1216370.1216371

Mohammed Gharib received the B.S. degree from Baghdad university of Technology, Iraq, in 2007 and M.S. degree from the Sharif University of Technology, Tehran, Iran in 2009. In September 2010, he joined performance and dependability laboratory (PDL) where he is working toward his PhD degree in computer engineering department at Sharif University of Technology, Tehran, Iran. During the year 2014, he was a visiting research scholar in California Institute for Telecommunications and Information Technology, University of California Irvine, Irvine. His research interests include mobile ad hoc networks, wireless sensor networks, peer-to-peer networks, and their security aspects.

Homayoun Yousefi'zadeh received E.E.E and Ph.D. degrees from the Dept. of EE-Systems at USC in 1995 and 1997, respectively. Currently, he is an Adjunct Professor at the Department of EECS at UC, Irvine. In the recent past, he was a Consulting Chief Technologist at the Boeing Company and the CTO of TierFleet. He is the inventor of several US patents, has published more than seventy scholarly reviewed articles, and authored more than twenty design articles associated with deployed industry products. Dr. Yousefi'zadeh is/was with the editorial board of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE COMMUNICATIONS LETTERS, IEEE Wireless Communications Magazine, IEEE JSTSP, and Journal of Communications Networks. He was the founding Chairperson of systems' management workgroup of the Storage Networking Industry Association and a member of the scientific advisory board of Integrated Media Services Center at USC. He is a Senior Member of the IEEE and the recipient of multiple best paper, faculty, and engineering excellence awards.

Ali Movaghar is a professor in the department of Computer Engineering at Sharif University of Technology in Tehran, Iran. He received his B.S. degree in Electrical Engineering from the University of Tehran in 1977, and M.S. and Ph.D. degrees in Computer, Information and Control Engineering from the University of Michigan, Ann Arbor, in 1979 and 1985, respectively. He visited the Institute National de Recherche en Informatique et en Automatique in Paris, France and the department of Electrical Engineering and Computer Science at the University of California, Irvine in 1984 and 2011, worked at AT&T Information Systems in Napervile, IL in 1985-1986, and taught at the University of Michigan, Ann Arbor in 1987-1989. His research interests include performance/dependability modeling and formal verification of wireless networks and distributed real-time systems. He is a Senior Member of the IEEE and the ACM.