Robust Confidentiality Preserving Data Delivery in Federated Coalition Networks

Lu Su*, Yunlong Gao†, Fan Ye‡, Peng Liu¶, Oktay Gunluk§, Tom Berman§, Seraphin Calo§, Tarek Abdelzaher†
*State University of New York at Buffalo, †University of Illinois at Urbana-Champaign
‡Peking University, ¶The Pennsylvania State University, §IBM T. J. Watson Research Center
lusu@buffalo.edu, {ygao12, zaher}@illinois.edu, yefan@pku.edu.cn, pliu@ist.psu.edu, {gunluk, scalo}@us.ibm.com

***Abstract*—Federated coalition networks are formed by interconnected nodes belonging to different *friendly-but-curious* parties cooperating for common objectives. Each party has its policy regarding what information may be accessed by which other parties. Data delivery in coalition networks must provide both confidentiality and robustness. First, data should remain confidential when passing through intermediate nodes belonging to parties not authorized to see its content. Second, data delivery has to be robust against dynamic topology changes caused by frequent node churn and failures. We utilize the technique of linear network coding to transform the original data into multiple coded packets and send them along different paths in a way such that no other party can reconstruct the data. This lightweight approach provides confidentiality and robustness for friendly-but-curious coalitions with much less complexity than cryptography methods. In addition, we formulate an optimization problem to find minimum-cost paths, and use column generation framework to address the huge number of variables. Based on the proposed algorithms, we develop a Robust Confidentiality Preserving (R-CP) data delivery protocol. Our evaluation demonstrates that the proposed method can find the optimum solution in several seconds for networks of a few thousands nodes, and deliver data at a high success rate.**

I. Introduction

Federated coalition networks [1]–[3] are heterogeneous distributed systems formed by interconnected nodes owned by different parties. Each party is an independent entity, such as an organization or a company that wants to pursue common objectives only possible by cooperating with other parties. Federated coalition networks are widely applied in various scenarios ranging from business or scientific collaborations to multinational humanitarian actions. For example, in a humanitarian emergency cooperation, government agencies and non-governmental organizations (NGOs) from possibly different countries work together to provide disaster relief and recovery. In such applications, each coalition party operates and maintains a certain number of computing and communicating devices that establish connections among themselves. The communication between two nodes usually has to pass through nodes owned by third parties.

Federated coalition networks are formed based on certain mutual trust, and coalition parties are usually *friendly-but-curious*: they play by pre-agreed rules and do not intentionally violate the rules (e.g., launching attacks). For example, nodes operated by the Red Cross do not attack those by another rescue team. However, they may read contents of the packets in transit through their nodes.

Data delivery in coalition networks has to provide data-confidentiality and robustness simultaneously. First, each party has its policies regarding what kinds of information may be accessed by which other parties. Not all information can be seen by everybody. If two nodes need to exchange some private data (e.g., personal information of team members, internal organization or planning information), they cannot send it as-is because the data may have to go through intermediate nodes belonging to other parties that are not authorized to see it. Second, the topology of a coalition network could be highly dynamic. Possible reasons include node mobility, the join or leave of individual nodes or entire parties, and node failures caused by various issues such as severe weather conditions. The data delivery mechanism needs to ensure robustness against these factors.

A suitable solution has to address both challenges. Traditionally, data confidentiality is guaranteed by end-to-end encryption, coding the original data with secret keys such that nobody can derive the plain text without knowing the key. Robustness in highly dynamic networks, on the other hand, is usually achieved by multi-path delivery. Thus combining data encryption and multipath delivery will satisfy both requirements.

Nevertheless, we note that cryptography methods are designed to handle much stronger attacks (e.g., malicious violation of rules), whereas in a *friendly-but-curious* environment, the concern is mainly to avoid the disclosure of information to unintended parties. In this paper, we explore an alternative, lightweight approach: use linear network coding [4], [5] and deliver the coded packets along multiple paths to achieve the dual goal of confidentiality and robustness. Specifically, a source node first partitions the original data into k chunks, and generates m ($m \geq k$) coded packets each of which is a different linear combination of the k chunks. The coded packets are transported to the destination through multiple paths. As long as the destination node can receive at least k of the coded packets, it can reconstruct the original data. To ensure confidentiality, the paths have to be chosen carefully such that no party not authorized to see the data can observe k or more coded packets.

Compared to encryption-based methods, this approach circumvents the issues those methods have to face in federated coalition networks as well as many other distributed networking systems [6]–[8]. First, due to the lack of a common authority, it is difficult to deploy public key infrastructure because no party can host a central certificate authority (CA). Second, since nodes or even parties may join and leave the network frequently, effective key management requires signif-

‡Work completed at IBM Research.

¶Peng Liu was supported by U.S. ARL and U. K. MoD W911NF-06-3-0001 and NSF CNS-0916469.

ISBN 978-3-901882-58-6

icant efforts and complexity. Finally, some node may possess limited resources ineffective of carrying out computationally expensive cryptography algorithms.

Although the proposed approach does not guarantee *perfect security* targeted by cryptography methods, which withstand intentional attacks, it provides *sufficient confidentiality* under the friendly-but-curious model of coalition networks. It is computationally light weight, key-management free, and it naturally incorporates multipath for robust delivery. Thus we believe it is worthwhile to explore alternatives that present a different set of tradeoffs. They would provide a valuable "mirror for reflection" for cryptography research efforts [9], [10] that aim to address the above issues.

In particular, we make the following contributions:

First, we propose an alternative approach of linear network coding together with multipath delivery to solve the robust confidentiality preserving data delivery problem in coalition networks. Compared to traditional encryption based approaches, it avoids complexities in addressing issues such as lack of common authority, expensive key management, and frequent node churn posed by coalition networks. This is possible thanks to the unique friendly-but-curious model in a coalition network where parties do not launch intentional attacks.

Second, we further optimize the selection of the paths used for delivering coded packets so as to not only satisfy confidentiality, but also minimize the costs of communication and computation required for the nodes on these paths. We formulate this problem as an integer programming problem, and relax it into a linear programming problem that has much less algorithmic complexity. To address the huge number of variables in the subsequent linear problem, we employ column generation [11], a classic technique to solve linear programs with huge number of variables. Finally, we develop a branch-and-price method to derive integral solutions from the solutions of the linear problem.

Third, based on the proposed algorithms, we develop a Robust Confidentiality Preserving (RCP) data delivery protocol that can adapt to different types of coalition networks (i.e., wired, wireless, or hybrid network). It makes use of the topology information collected by the underlying routing protocols used to transport plain text data allowed to be seen by all the parties, and results in little additional control overhead. We compared our scheme to a naive baseline method under a wide variety of coalition network settings, and found that it runs two orders of magnitude faster while still achieving optimality. Even in networks of 3000 nodes owned by 10 parties, the computation finishes in several seconds, making it feasible for large coalition environments.

II. Problem Overview

In this section, we first provide a detailed description of the problem we target on. Then, we introduce the threat model and system model applied in this paper.

A. Problem Description

In a coalition network, nodes are maintained and managed by different parties. As shown in Fig. 1, the nodes are marked by different colors, each of which corresponds to a particular party. Suppose the source node s wants to send some data to the destination node t. Assume the original data is partitioned into k chunks, which are used to generate $m \geq k$ coded packets through linear network coding [4], [5]. The m coded packets are then sent to the network through different paths. The destination node t can recover the original data as long as it collects at least k out of these m packets. If a packet goes through a node in the network, we say that this packet is observed by the party who owns this node. Our objective is to find a path for each packet such that the aggregate computation and communication cost of the m packets can be minimized, and meanwhile, no parties (colors) that are not authorized to see the data can observe (cover) more than $k-1$ packets. In the rest of this paper, to simplify the presentation, we assume the parties other than the ones owning the source and destination nodes do not have authorized access to the data.

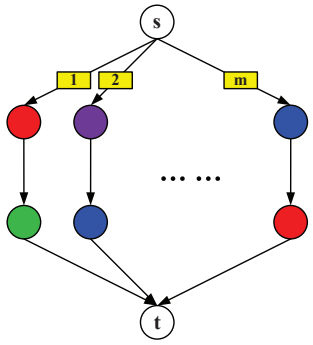


Fig. 1. An example of coalition network

B. Threat Model

Regarding the friendly-but-curious threat model, we make the following assumptions: (1) There is no attack other than eavesdropping. (2) A party can do eavesdropping in arbitrary ways on its own machines/devices, but not on those owned by other parties.

These assumptions are due to the unique characteristics of federated coalition networks. A key distinction of coalition networks is that they will not be formed in the first place without sufficient pre-existing *mutual trust* among the parties. Such mutual trust implies that no party is malicious, even though it may still be curious about data owned by other parties.

C. System Model

In the next section, we will formulate the above problem as an integer program. Before getting into the details of the mathematical formulation, we first introduce some notations and terminologies.

First of all, the network is modeled as a directed graph $G_0(V_0, E_0)$, where V_0 and E_0 are the sets of nodes and links, respectively. In this graph, each link $(u, v) \in E_0$ is associated with a positive (integer) cost c_{uv}. The cost represents the resources (e.g., communication energy, latency time) consumed when transmitting a packet along this link. For each node $u \in V_0$, let $p_u \in \{1, \ldots, l\}$ denote the id of the party that u belongs to. Moreover, we use $E_{in}(u)$ and $E_{out}(u)$ to denote the sets of its incoming and outgoing links, respectively.

To further model the cost (e.g., computational energy and delay) consumed at each node, we transform the original graph into a new graph $G(V, E)$. An example of graph transformation is shown in Fig. 2.

The details of the transformation can be summarized as follows. First, for each node $u \in V_0$, we split it into two nodes

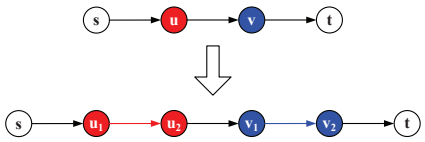


Fig. 2. An example of graph transformation

$u_1, u_2 \in V$, and connect u_1 and u_2 using a link $(u_1, u_2) \in E$. (u_1, u_2) is assigned a positive (integer) value $c_{u_1u_2}$ that represents the cost of u. Second, we connect all the incoming links of u in G_0 to u_1, and all the outgoing links of u in G_0 to u_2. In other words, we let $E_{in}(u_1) = E_{in}(u)$, $E_{out}(u_1) = \{(u_1, u_2)\}$, $E_{in}(u_2) = \{(u_1, u_2)\}$, and $E_{out}(u_2) = E_{out}(u)$. Third, we assign a party id to each of the links in E. For each link $(u, v) \in E_0$ in the original graph, let $p_{uv} = 0$. For link $(u_1, u_2) \in E - E_0$. let $p_{u_1u_2} = p_u$. For example, as shown in Fig. 2, the links (u_1, u_2) and (v_1, v_2) in the transformed graph bear the same party color in the original graph. In some applications of federated coalition networks, some links could be wireless. If (u, v) is a wireless link, due to the broadcast nature of wireless transmission, the packets sent by u may be overheard by some other nodes whose parties are not permitted to see the data. To model this scenario, we assign multiple party ids to (u, v). In particular, we let $p_{u_1u_2}$ be the set of parties whose nodes can hear the packets sent by u (including u's own party). Finally, we denote by $E_j (j = 1, \ldots, l)$ the set of links whose party set contains j-th party, and as a result we have $E = \bigcup_{j=0}^{l} E_j$.

III. Mathematical Formulation

With the previously defined notations and terminologies, we formulate the target problem as an integer program on the transformed graph. In this section, we first introduce the variables and constants involved in the integer program, then give detailed descriptions on the objective function as well as the constraints.

A. Variables and Constants

Variables: Between the source node s and the destination node t, we define a non-negative integer variable $x_p \in \mathbb{Z}_+$ on each path $p \in \mathcal{P}$, where $\mathcal{P}$ is the set of all the s-t paths. x_p denotes the number of different coded packets transported along path p.

Constants: Besides the aforementioned constants including the link cost c_{uv} and party id p_{uv}, we further define a couple of binary constants to model the relationship between path and party (link). The first constant, denoted by a_j^p, indicates whether the packets transported along path p can be observed by j-th party. It can be determined via checking the party set of each link on the path. The second constant, denoted by b_{uv}^p, indicates whether a link (u, v) is on path p. Finally, we use c_p to denote the cost of path p. Clearly, we have $c_p = \sum_{(u,v)\in p} c_{uv} = \sum_{(u,v)\in E} b_{uv}^p c_{uv}$.

B. Objective Function and Constraints

Given the variables and constants defined above, we give the formal formulation of the objective function as well as constraints.

Objective function: In this problem, we aims at minimizing the overall cost of the packet transportation. As previously discussed, the cost includes the resource consumption at both nodes and links, and can be modeled as positive values associated with the links of the transformed graph. Therefore, the objective function is given as follows:

$$\min \quad \sum_{p\in\mathcal{P}} c_p x_p = \sum_{p\in\mathcal{P}} x_p \sum_{(u,v)\in p} c_{uv}$$

Constraints: Next, we elaborate on the constraints that our objective function is subject to.

1) Packet delivery constraint: Mathematically, we want to ensure that all the coded packets can be delivered to the destination. This can be achieved via the following equality:

$$\sum_{p\in\mathcal{P}} x_p = m$$

In practice, the packets could be lost due to topology change. However, the proposed network coding based approach generates redundant coded packets and the destination node can recover the original data as long as it collects enough number of packets.

2) Packet confidentiality constraint: As we previously discussed, each party other than the ones that own the source and destination cannot obtain more than $k-1$ coded packets, otherwise it will be able to recover the original data. Our path-centric formulation makes it easy to model this constraint:

$$\sum_{p\in\mathcal{P}} a_j^p x_p \le k-1 \quad \forall\, j \in \{1, \ldots, l\}$$

C. Optimization Program

Putting everything together, we have the following optimization program:

$$\begin{aligned} \mathbf{P}: \quad z^* = \min \quad & \sum_{p\in\mathcal{P}} c_p x_p \\ \text{s.t.} \quad & \sum_{p\in\mathcal{P}} a_j^p x_p \le k-1 \quad \forall\, j \in \{1, \ldots, l\} \\ & \sum_{p\in\mathcal{P}} x_p = m \\ & x_p \in \mathbb{Z}_+ \end{aligned}$$

The difficulty of solving $\mathbf{P}$ is twofold. First, $\mathbf{P}$ is an integer problem, which in general is hard to solve in polynomial time. Second, since each variable x_p corresponds to a path, the number of variables potentially can be very large, especially when the network size is huge. To address these challenges, we first relax $\mathbf{P}$ into a linear program, then apply column generation [11], a classic technique to deal with huge number of variables, on the relaxed program. In order to derive integral solutions from the solutions of the relaxed program, we adopt a searching strategy called branch-and-price [12]. In the following, we will first introduce the column generation framework used to solve the relaxed problem in Section IV, then explain in depth the branch-and-price strategy used to find integral solutions in Section V.

IV. Column Generation

Column generation is designed to solve linear programs with too large numbers of variables. It extends simplex [13], the most widely used linear programming algorithm. Simplex effectively maintains a feasible solution consisting of some variables taking non-zero values (called *basic variables*) at all times, and iteratively updates the solution by replacing a basic variable with a non-basic one (the variable with zero value). Each non-basic variable has a *reduced cost* whose

value, if negative, indicates how much the objective function can improve when this variable replaces an existing basic variable. Simplex terminates with the optimal solution when no non-basic variable with negative cost can be found.

The simplex method has an obvious problem, that is, when the number of variables is large, it is difficult to maintain them in an explicit way (for example, in a tableau where each column records the constraint coefficients associated with a variable). Column generation addresses this problem through exploiting one feature of the optimal solution: most of the variables will be non-basic variables with zero value, and only a small subset of variables need to be considered when solving the problem. Column generation leverages this idea to generate only the variables (more precisely, the coefficient columns of variables) which have the potential to improve the objective function. Specifically, it reduces the original linear problem (called master problem, denoted by $\mathbf{MP}$) into a smaller problem (called restricted master problem, denoted by $\mathbf{RMP}$) where only the columns corresponding to basic variables are maintained. In each iteration, a new optimization problem (called subproblem) is created to identify the non-basic variable with the lowest reduced cost. This variable is then added to the restricted master problem, which is solved again. The process is repeated until no negative reduced cost variables are identified. In other words, when the subproblem returns a solution with non-negative reduced cost, we can conclude that the solution to the original master problem is optimal.

In following subsections, we elaborate in detail on how the column generation algorithm is applied to solve our problem.

A. Master Problem

In this case, the master problem is the linear relaxation of the original program $\mathbf{P}$:

$$\mathbf{MP}: \quad z^*_{\mathbf{MP}} = \min \quad \sum_{p\in\mathcal{P}} c_p x_p \tag{1}$$

$$\text{s.t.} \quad \sum_{p\in\mathcal{P}} a_j^p x_p \le k-1 \;\forall\; j \in \{1,\dots,l\} \tag{2}$$

$$\sum_{p\in\mathcal{P}} x_p = m \tag{3}$$

$$x_p \ge 0 \tag{4}$$

We use the simplex method to solve $\mathbf{MP}$. In each iteration of the simplex method we look for a non-basic path flow variable x_p to enter the basis. This explicit pricing (i.e., selection of non-basic variable) is a too costly operation when the number of variables (paths) is huge, since the simplex tableau maintains a column for each variable. Instead, we work with the restricted master problem ($\mathbf{RMP}$), which takes as input a reasonably small subset of paths $\mathcal{P}_{\mathbf{RMP}} \subset \mathcal{P}$. The reduced costs of the remaining paths are evaluated by implicit enumeration and the path with the most negative reduced cost is added to $\mathcal{P}_{\mathbf{RMP}}$. In the next two subsections, we first explain how the reduced cost of a path is derived, and then formulate a subproblem to identify the path with the lowest reduced cost.

B. Reduced Cost

We associate each constraint of $\mathbf{MP}$ with a Lagrange multiplier. The Lagrange multipliers can be interpreted as the "shadow prices" of the constraints, which can be understood as the "costs" to be charged if the constraints are violated. In particular, we define a party price π_j $(j = 1,\dots,l)$ for each packet confidentiality constraint (Eqn. (2)) and a packet price π_0 for the packet delivery constraint (Eqn. (3)). Then, we derive the Lagrangian of $\mathbf{MP}$ as follows:

$$\begin{aligned}
&L(\mathbf{x},\boldsymbol{\pi}) \\
&= \sum_{p\in\mathcal{P}} c_p x_p + \sum_{j=1}^{l} \pi_j \sum_{p\in\mathcal{P}} a_j^p x_p - \sum_{j=1}^{l} \pi_j (k-1) - \pi_0 \sum_{p\in\mathcal{P}} x_p + \pi_0 m \\
&= \sum_{p\in\mathcal{P}} c_p x_p + \sum_{p\in\mathcal{P}} \sum_{j=1}^{l} \pi_j a_j^p x_p - \sum_{p\in\mathcal{P}} \pi_0 x_p + \pi_0 m - \sum_{j=1}^{l} \pi_j (k-1) \\
&= \sum_{p\in\mathcal{P}} (c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0) x_p + \pi_0 m - \sum_{j=1}^{l} \pi_j (k-1)
\end{aligned}$$

For each path flow variable x_p, the reduced cost is its coefficient in the Lagrangian:

$$c_p^\pi = c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0 \tag{5}$$

The path flow variables x_p are optimal for $\mathbf{MP}$ if and only if for the party prices π_j $(j = 1,\dots,l)$ and the packet price π_0, the reduced costs and the path flows satisfy the following conditions:

- Dual feasibility condition:

$$c_p^\pi \ge 0, \quad \forall\; p \in \mathcal{P}$$

- Dual complementary slackness condition:

$$\pi_j (\sum_{p\in\mathcal{P}} a_j^p x_p - k + 1) = 0, \quad j = 1,\dots,l$$

- Primal complementary slackness condition:

$$x_p c_p^\pi = 0, \quad \forall\; p \in \mathcal{P}$$

These optimality conditions have intuitive interpretations. Specifically, for a path p along which some packets are delivered, we have $x_p > 0$. According to the primal complementary condition, the reduced cost is

$$c_p^\pi = c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0 = 0$$

Namely, the modified cost of this path (i.e., $c_p + \sum_{j=1}^{l} \pi_j a_j^p$) is equal to π_0. Furthermore, since $c_p^\pi \ge 0$ for all the paths, π_0 can be regarded as a shortest path distance that will be discussed in the next subsection.

C. Subproblem

At each iteration of the simplex algorithm, we get an optimal dual solution $\boldsymbol{\pi} = (\pi_0, \pi_1, \dots, \pi_l)$ of $\mathbf{RMP}$, and shall choose the path flow variable x_p (recall that each path flow variable corresponds to a column in the simplex tableau) with the minimum reduced cost $c_p^{\pi *} = c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0$ to enter the $\mathbf{RMP}$. A native solution is to explicitly calculate the reduced cost of each path, and pick the optimal one. In practice, however, this is not feasible given the huge number of paths. To address this challenge, the column generation technique suggests to evaluate the reduced costs implicitly, in other words, to identify the optimal path through solving

a pricing subproblem, which is written as:

$$\mathbf{SP}: \quad z^*_{\mathbf{SP}} = \min \quad c_p + \sum_{j=1}^{l} \pi_j a_j^p$$
$$\text{s.t.} \quad p \in \mathcal{P}$$

SP cannot be solved directly, and thus we transform it into a link-centric version. Specifically, we define a binary variable x_{uv} indicating whether the link (u, v) is on the selected path with the minimum reduced cost, and an auxiliary binary variable y_j indicating whether the selected path passes a node owned by the j-th party. Then we have an equivalent optimization program:

$$\min \quad \sum_{(u,v)\in E} c_{uv} x_{uv} + \sum_{j=1}^{l} \pi_j y_j \tag{6}$$

$$\text{s.t.} \quad \sum_{(u,v)\in E} x_{uv} - \sum_{(v,u)\in E} x_{vu} = \begin{cases} 1 & \text{if } u = s \\ -1 & \text{if } u = t \\ 0 & \text{otherwise} \end{cases} \quad \forall\, u \in V \tag{7}$$

$$y_j = \max_{(u,v)\in E_j} x_{uv} \quad \forall\, j \in \{1, \ldots, l\} \tag{8}$$

$$x_{uv} \in \{0, 1\}, \quad y_j \in \{0, 1\} \tag{9}$$

The difficulty of solving this program lies in the constraint Eqn. (8) that has a max function. Since y_j has a positive coefficient in the objective function which is to be minimized, the constraint Eqn. (8) can be replaced by $y_j \geq x_{uv}$.

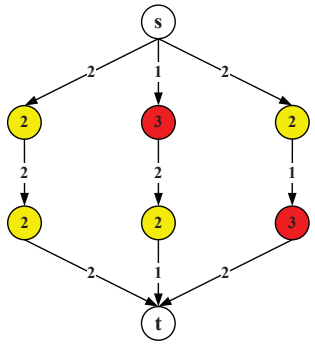


Fig. 3. An example of shortest coalition path

In fact, this problem is to find a shortest coalition path with the minimum reduced cost. However, it is different from the traditional shortest path problem in which the cost of a path is the summation of the edges' or vertices' weights. A simple example is shown in Fig. 3. In the figure, the number on each edge denotes the cost of this edge (i.e., c_{uv}), while the number on each node implies the price of the party which owns this node (i.e., π_j)[1]. From left to right, we denote the three paths by p_1, p_2 and p_3. As can be seen, if we simply add up all the numbers on edges and nodes along each path, p_2 has the smallest cost, which is 9. However, since along p_1 there is only one party, its price should be counted only once. Thus, the actual cost of p_1 should be 8, which is the smallest.

As a matter of fact, the shortest coalition path problem is an NP-hard problem. We prove this proposition by the following theorem:

[1]For the simplicity of presentation, here we assume there is no cost for each node, and associate parties with nodes. In actual implementation, the parties are associated with links in the transformed graph, as described in Section II.

Theorem 1: The shortest coalition path problem is NP-hard.

Proof: In this proof, we show that the set cover problem (SCP), which is known as a NP-complete problem, can be reduced to the shortest coalition path problem (CPP). The reduction algorithm begins with an instance of set cover problem. Given a set of elements $\mathcal{U} = \{1, 2, \ldots, m\}$ (called the universe) and a set $\mathcal{S}$ of n sets whose union equals the universe, the set cover problem is to identify the smallest subset of $\mathcal{S}$ whose union contains all elements in the universe. Based on the instance of SCP, an instance of CPP can be constructed through building up the correspondence between party and element set. Specifically, for each element $i \in \mathcal{U}$, suppose it is covered by n_i different sets, we create n_i different nodes, each of which corresponds to a set (party). Each node of element i is connected to each node of element $i + 1$ via a directed edge. The source node s is connected to the nodes of element 1, while element m's nodes link to the destination node t.

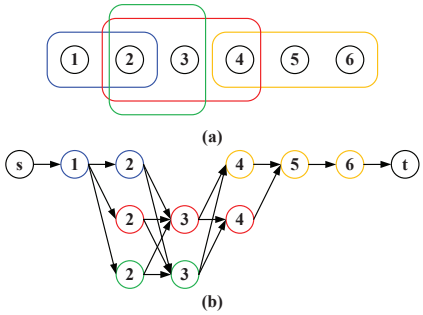


Fig. 4. NP-hardness of shortest coalition path

Fig. 4 shows a simple instance of SCP and the CPP instance constructed based on it. In this case, 6 elements are covered collaboratively by 4 different sets with different colors. The elements (e.g., element 2) covered by multiple sets are transformed into multiple nodes in the CPP instance. It is clear that a solution (the subset of $\mathcal{S}$ covering all the elements) to the SCP instance corresponds to at least one solution (a path passing all the nodes from s to t covered by a subset of parties) to the CPP instance. Therefore, a SCP solution is optimal if and only if its counterpart in CPP is optimal. The reduction takes $O(mn)$ time and thus CPP is NP-hard. ■

Though in general the shortest coalition problem is NP-hard, if the number of different parties (colors) is small and thus can be treated as a constant, we can use a polynomial-time algorithm to find the optimal solution. The basic idea of this algorithm is as follows. We exhaustively check all the subsets of parties (colors). For each particular party subset, we remove from the original graph the nodes (together with the links attached to these nodes) whose parties are not in this subset. In the resultant subgraph, we find the shortest path with regard only to the link cost c_{uv}. After we collect the shortest paths of all the party subsets, we calculate their reduced costs, and pick the one with the lowest reduced cost. The proof of the algorithm's correctness is straightforward and thus omitted here.

When $c_p^{\pi*} \geq 0$ (i.e., $z^*_{\mathbf{SP}} \geq \pi_0$), there is no negative reduced cost column, and the algorithm terminates. When a $c_p^{\pi*} < 0$, we add the column corresponding to the shortest coalition path, which is $[\ c_p, \quad a_1^p, \ldots, a_l^p, \quad 1\ ]^{\mathrm{T}}$, to the **RMP**.

D. Lower Bound

It is important to note that it can take a long time to solve the master problem ($\mathbf{MP}$) to optimality. In practice, we do not need an optimal solution [14]. It is enough to find a near-optimal solution, i.e., a primal solution that comes close enough to the optimal cost of $\mathbf{MP}$ (i.e., $z^*_{\mathbf{MP}}$). To prove that our solution is near-optimal, we need to derive a lower bound on $z^*_{\mathbf{MP}}$. This can be achieved through exploiting the Lagrangian dual problem of $\mathbf{MP}$:

$$\begin{aligned}\mathbf{MD}: \quad z^*_{\mathbf{MD}} &= \max_{\boldsymbol{\pi}} \min_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\pi}) \\ &= \max_{\boldsymbol{\pi}} \min_{\mathbf{x}} \Big\{ \sum_{p\in\mathcal{P}} (c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0) x_p \\ &\quad + \pi_0 m - \sum_{j=1}^{l} \pi_j (k-1) \Big\}\end{aligned}$$

According to the primal complementary slackness condition, when optimality is attained, $\sum_{p\in\mathcal{P}}(c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0)x_p = 0$, and thus $z^*_{\mathbf{MP}} = z^*_{\mathbf{MD}} = \pi_0 m - \sum_{j=1}^{l} \pi_j (k-1)$. Furthermore, given a vector of Lagrange multipliers $\boldsymbol{\pi} = (\pi_0, \pi_1, \ldots, \pi_l)$, the dual objective value is

$$\begin{aligned}\min_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\pi}) &= \min_{\mathbf{x}} \Big\{ \sum_{p\in\mathcal{P}} (c_p + \sum_{j=1}^{l} \pi_j a_j^p - \pi_0) x_p \Big\} \\ &\quad + \pi_0 m - \sum_{j=1}^{l} \pi_j (k-1) \\ &= mc_p^{\pi *} + z^*_{\mathbf{MP}}\end{aligned}$$

The same rules hold for the restricted master program ($\mathbf{RMP}$). Therefore, the optimal objective value of the master problem is bounded by:

$$mc_p^{\pi *} + z^*_{\mathbf{RMP}} \le z^*_{\mathbf{MP}} \le z^*_{\mathbf{RMP}}$$

Here $mc_p^{\pi *} + z^*_{\mathbf{RMP}}$ is the objective value of $\mathbf{MD}$ (also the dual objective value of $\mathbf{MP}$), given the optimal dual variables of $\mathbf{RMP}$. Since $z^*_{\mathbf{MP}} \le z^*$ (the optimal value of the original integer program $\mathbf{P}$), $mc_p^{\pi *} + z^*_{\mathbf{RMP}}$ is also a lower bound on z^*. In general, $z^*_{\mathbf{RMP}}$ is not a valid upper bound on z^*, except if the current $\mathbf{x}$ variables are integer. The algorithm is exact and finite as long as finiteness is ensured in optimizing the $\mathbf{RMP}$.

E. The algorithm

In this subsection, we present the column generation algorithm. Earlier subsections have discussed the major parts of the algorithm. Here we turn these parts into a whole to provide a global picture of column generation. Basically, the detailed steps of the column generation algorithm are shown below:

1) Initialization: Select $n_0 < n$ paths between the source and destination (corresponding to n_0 columns) to the $\mathbf{RMP}$.

2) Solve the restricted master problem: Solve $\mathbf{RMP}$ and get the optimal dual solution $\boldsymbol{\pi}$.

3) Solve the pricing subproblem: Solve the $\mathbf{SP}$ problem and find the shortest coalition path with $c_p^{\pi *} < 0$. The solution found corresponds to a variable x_p of the master problem with negative reduced cost, and thus is a candidate to enter the basis of the current restricted master problem.

4) Optimality test: If $c_p^{\pi *} \ge 0$, then no master problem variable prices out negative to enter the basis of $\mathbf{RMP}$. The current restricted master problem solution is an optimal solution to the complete master problem.

5) Near-optimality test: If the $\mathbf{RMP}$ is feasible, and the pricing subproblem is feasible and bounded. Calculate the lower and upper bounds of $z^*_{\mathbf{MP}}$

$$mc_p^{\pi *} + z^*_{\mathbf{RMP}} \le z^*_{\mathbf{MP}} \le z^*_{\mathbf{RMP}}$$

If their gap is within user specified optimality tolerances, then the current restricted master problem solution solves the master problem to the requested tolerance.

6) Update the restricted master problem: As the optimality test failed, at least one master problem variable was identified when solving the pricing subproblem that is a candidate to enter the basis of $\mathbf{RMP}$. Choose at least one of these to add to the current $\mathbf{RMP}$. On the other hand, any current non-basic variable of the $\mathbf{RMP}$ with positive reduced costs may be removed. Then, the algorithm will go back to step 2 to solve the updated $\mathbf{RMP}$. Here we should note that it is not necessary to rebuild the simplex tableau of $\mathbf{RMP}$ from scratch. What the algorithm does is to add a column to the current tableau, and continue the simplex iteration.

Since column generation is a specialised simplex method, it inherits the problem of simplex method that cannot guarantee a polynomial time computational complexity in worst case. However, like simplex, it is remarkably efficient in practice.

V. Finding Integral Solutions

The column generation approach does not automatically guarantee integer solutions. If when the column generation algorithm terminates, some elements of the optimal solution vector are not integers, then the original problem $\mathbf{P}$ is not yet solved. To address this problem, we propose to use branch-and-price [12], a widely used technique associated with column generation to find integral solutions. In our case, we branch on the link flow denoted by $f_{uv} = b_{uv}^p x_p$ (recall that b_{uv}^p indicates whether a link (u, v) is on path p). To achieve this, we add lower and upper bounds to the link flow in the master problem $\mathbf{MP}$:

$$\begin{aligned}\mathbf{MP}: \quad z^*_{\mathbf{MP}} = \min \quad & \sum_{p\in\mathcal{P}} c_p x_p \\ \text{s.t.} \quad & \sum_{p\in\mathcal{P}} a_j^p x_p \le k-1 \quad \forall\, j \in \{1, \ldots, l\} \\ & \sum_{p\in\mathcal{P}} x_p = m \\ & \sum_{p\in\mathcal{P}} b_{uv}^p x_p \le u_{uv} \quad \forall\, (u,v) \in E \\ & \sum_{p\in\mathcal{P}} b_{uv}^p x_p \ge l_{uv} \quad \forall\, (u,v) \in E \\ & x_p \ge 0\end{aligned}$$

The addition of the constraints would result in changes in the pricing subproblem. In particular, the Lagrangian of the modified $\mathbf{MP}$ is

$$\begin{aligned}&L(\mathbf{x}, \boldsymbol{\pi}) \\ &= \sum_{p\in\mathcal{P}} c_p x_p + \sum_{j=1}^{l} \pi_j (\sum_{p\in\mathcal{P}} a_j^p x_p - k + 1) - \pi_0 (\sum_{p\in\mathcal{P}} x_p - m) \\ &\quad + \sum_{(u,v)\in E} \pi_{uv}^u (\sum_{p\in\mathcal{P}} b_{uv}^p x_p - u_{uv}) - \sum_{(u,v)\in E} \pi_{uv}^l (\sum_{p\in\mathcal{P}} b_{uv}^p x_p - l_{uv})\end{aligned}$$

$$= \sum_{p\in\mathcal{P}}(c_p + \sum_{j=1}^{l}\pi_j a_j^p + \sum_{(u,v)\in E}\pi_{uv}^u b_{uv}^p - \sum_{(u,v)\in E}\pi_{uv}^l b_{uv}^p - \pi_0)x_p$$

$$+ \pi_0 m - \sum_{j=1}^{l}\pi_j(k-1) - \sum_{(u,v)\in E}\pi_{uv}^u u_{uv} + \sum_{(u,v)\in E}\pi_{uv}^l l_{uv}$$

In this case, the reduced cost of each path p is

$$c_p^\pi = c_p + \sum_{j=1}^{l}\pi_j a_j^p + \sum_{(u,v)\in E}\pi_{uv}^u b_{uv}^p - \sum_{(u,v)\in E}\pi_{uv}^l b_{uv}^p - \pi_0$$

$$= \sum_{(u,v)\in E} b_{uv}^p(c_{uv} + \pi_{uv}^u - \pi_{uv}^l) + \sum_{j=1}^{l}\pi_j a_j^p - \pi_0$$

$$= \sum_{(u,v)\in p}(c_{uv} + \pi_{uv}^u - \pi_{uv}^l) + \sum_{j=1}^{l}\pi_j a_j^p - \pi_0$$

Thus, the objective function of the corresponding subproblem **SP** becomes $\sum_{(u,v)\in E}(c_{uv} + \pi_{uv}^u - \pi_{uv}^l)x_{uv} + \sum_{j=1}^{l}\pi_j$.

The detailed steps of the branch-and-price algorithm are shown below:

1) Initialization: Initially, we set u_{uv} and l_{uv} to be $+\infty$ (it could also be the link capacity) and 0, and thus the modified **MP** becomes the same as the original one.

2) Branch: At each branch, we calculate all the link flows, and pick the most fractional flow defined as below:

$$(\hat{u}, \hat{v}) = \arg\min_{(u,v)\in E} |(x_{uv} - \lfloor x_{uv} \rfloor) - 0.5|$$

Then, we create two child problems as follows. In the first child problem, we set $u_{\hat{u}\hat{v}} = \lfloor x_{\hat{u}\hat{v}} \rfloor$. On the other hand, let $l_{\hat{u}\hat{v}} = \lceil x_{\hat{u}\hat{v}} \rceil$ in the second child problem.

3) Feasibility test: If a child problem has no feasible solution, stop searching on this branch.

4) Worse solution: If a child problem has an optimal solution that is worse than the incumbent, i.e., the best feasible integer solution found so far, stop searching on this branch.

5) Better integer solution: If a child problem has an integer optimal solution that is better than the incumbent, update incumbent with this solution and stop searching on this branch.

6) Better fractional solution: If a child problem has an fractional optimal solution that is better than the incumbent, go to step 2 and further branch this problem.

The above process is repeated until all the link flows become integral. Like simplex, the branch-and-price method cannot guarantee polynomial-time complexity in worst case. Nevertheless, it works well in practice.

VI. Protocol Design

Based on the algorithms presented previously, we come up with the robust confidentiality preserving (RCP) data delivery protocol for federated coalition networks. As aforementioned, the federated coalition networks can be built on wired, wireless, or hybrid infrastructures, and thus the RCP protocol should be adaptive to any network genre. Towards this end, RCP makes the best use of the information provided by the underlying unicast/multicast/broadcast routing protocols used to transport the "plaintext" data (e.g., notifications of coalition-wide meetings) that are allowed to be seen by any party of the coalition. The routing protocols are distinct in different type of networks.

For the networks with relatively stable topologies, proactive routing protocols (e.g., link state routing) are usually used. Under such protocols, each node maintains a routing table indicating the next hop on the shortest path towards any other node in the network. To achieve this, individual nodes exchange neighbor information in order to have a map of the entire network. RCP takes advantage of the topology information stored in the nodes. The additional information needed by RCP is just the party id of each node, which is not difficult to obtain. RCP lets the source node carry out all the calculations based on its local topology map, and embeds the path for each coded packet into its header. The packets are routed to the destination node according to the path information in the header. Upon the loss of a coded packet, no retransmission is needed. By carefully selecting the parameters m and k, robust data delivery can be achieved with high probability.

For the networks with dynamic topologies, on-demand routing protocols (e.g., dynamic source routing) are favored. On-demand routing protocols are invoked only when the source node has some packets to transmit. It floods the route query information throughout the whole network, and then collects the path information returned from the destination. In order to reduce the overhead of route query messages, these routing protocols usually let each node cache the paths discovered recently. We can build a partial view of the network from the cached paths using the scheme presented in [15], and apply the proposed algorithm on this partial network. The derived paths, although not optimal, can still preserve confidentiality and robustness.

VII. Performance Evaluation

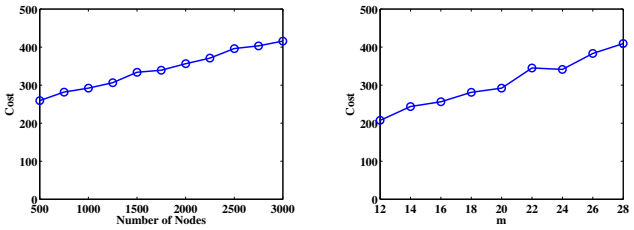
In this section, we evaluate the performance of the schemes proposed in the paper. We first introduce our experiment environment and settings, then present the evaluation results.

A. Experiment Environment and Settings

We conducted simulations of the proposed algorithms on a 64-bit Windows machine with 9GB RAM and Intel i7 CPU. In the simulations, we made use of JGraphT [16], an open source graph library, to generate random topologies of federated coalition networks, and implemented our column generation algorithm on CPLEX [17], an widely used optimization software package.

Unless otherwise stated, the default parameters are as follows. The network is organized by $l = 10$ coalition parties. The number of nodes in the network is 1000, and the average incoming and outgoing degree of nodes is 25 (to guarantee a connected topology). $m = 20$ coded packets are generated by the source node, and the data can be reconstructed by the destination node if it collects at least $k = 10$ coded packets.

B. Experiment Results



(a) Number of Nodes on Cost (b) Number of Packets on Cost

Fig. 5. Impact of number of nodes and number of packets on the cost

Cost: We first study the aggregate computation and communication cost of data delivery. In this experiment, we associate each link (node) with a randomly generated communication (computation) cost between 1 and 5. Figure 5 shows the impact of two parameters, i.e., number of nodes in the network and number of coded packets generated by the source, on the overall data delivery cost. As can be seen in Fig. 5(a), as the network size enlarges (with fixed node degree), the cost grows up due to the increase of path length. Intuitively, when more coded packets are injected into the network, the delivery cost would increase accordingly. This is confirmed by Fig. 5(b).

Computational Efficiency: The nodes in the federated coalition networks may possess limited resources and thus the data delivery algorithm should be computationally efficient. To evaluate computational efficiency of the proposed algorithm, we measure the time duration of executing the algorithm. For the purpose of comparison, we include a baseline scheme in the experiment. It results from another possible strategy of modeling the robust confidentiality preserving data delivery problem. Specifically, a binary variable $x_{uv}^i = \{0,1\}$ is defined for each coded packet $i \in \{1,\ldots,m\}$ and each link $(u,v) \in E$, indicating whether the i-th packet is passed through link (u,v). Similar to the subproblem **SP**, an auxiliary binary variable y_j^i is defined to denote whether the i-th packet passes through a node of j-th party. The complete problem formulation, denoted by **LP**, is listed as below:

$$\min \quad \sum_{(u,v)\in E} c_{uv} \sum_{i=1}^{m} x_{uv}^i$$

$$\text{s.t.} \quad \sum_{(u,v)\in E} x_{uv}^i - \sum_{(v,u)\in E} x_{vu}^i = \begin{cases} 1 & \text{if } u = s \\ -1 & \text{if } u = t \\ 0 & \text{otherwise} \end{cases}$$

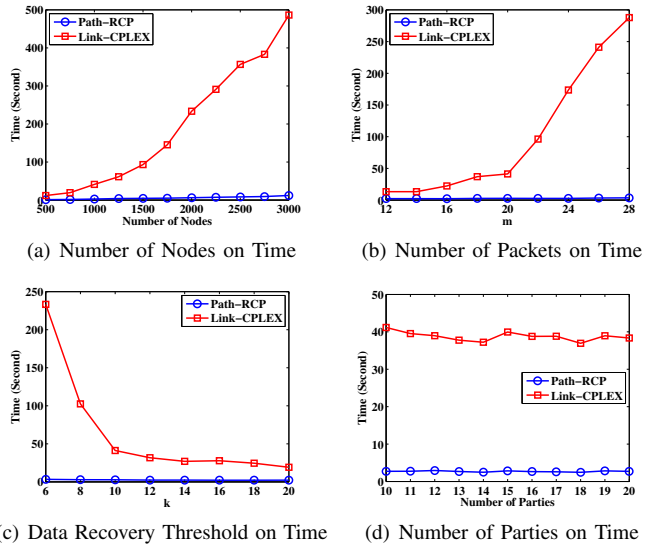
$$y_j^i \geq x_{uv}^i \tag{10}$$

$$\sum_{i=1}^{m} y_j^i \leq k-1 \tag{11}$$

$$x_{u,v}^i \in \{0,1\}, \quad y_j^i \in \{0,1\}$$

In **LP**, data confidentiality is preserved by the constraints (10) and (11). This link-centric formulation creates less variables than our path-centric formulation **P**, and can be directly fed to optimizers such as CPLEX. However, it has a problem of variable symmetry [18], since the variables can be permuted without changing the structure of the problem. This could result in much larger search space.

Figure 6 gives the comparison results of our RCP protocol designed to solve path-centric problem **P** and the baseline scheme that uses CPLEX to solve link-centric problem **LP** directly. The outputted objective values (i.e., the data delivery cost) of both schemes are always the same, and thus here we emphasize on the comparison of execution time. As one can see, under all the parameter settings, the proposed RCP protocol takes only several seconds, and thus can be applied on the coalition nodes whose computational capabilities are much weaker than the machine used to conduct this experiment. In contrast, it usually takes CPLEX two orders of magnitude more time to solve the link-centric problem **LP**. Here we would like to elaborate on Fig. 6(c) and Fig. 6(d), which are a little bit counter-intuitive. In Fig. 6(c), the execution time drops with the increase of k, the data recovery threshold. This is



(a) Number of Nodes on Time (b) Number of Packets on Time

(c) Data Recovery Threshold on Time (d) Number of Parties on Time

Fig. 6. Impact of various parameters on RCP's execution time

because the smaller k is, the less number of coded packets are allowed to be transported along a single path, and thus the more different paths are included in the solution. According to the laws of combinatorics, there are more combinations of choosing more paths from a large fixed path set, which implies a larger search space for the optimization problem. As shown in Fig. 6(d), the execution time does not change much under different numbers of parties. This implies that the algorithm we propose to solve the subproblem works very well in practice. Though the number of subgraphs (each corresponds to a subset of parties) increases as the number of parties grows, the size of each subgraph decreases. Therefore, the overall performance is constant regardless of the number of parties.

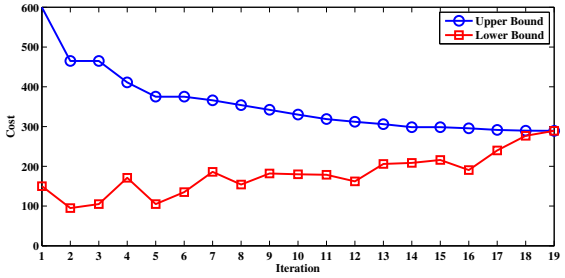


Fig. 7. Convergence of column generation

Convergence: Figure 7 demonstrates the convergence of the proposed algorithm. As discussed in Section IV-D, the upper and lower bounds of the optimal objective value, although oscillate, converge to the optimal value within a small number of iterations.

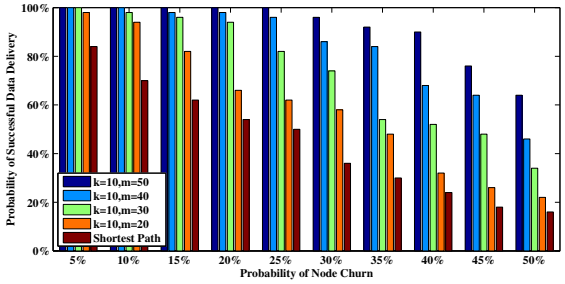


Fig. 8. Comparison of Robustness

Robustness: In federated coalition networks whose topologies

are dynamically changing (due to node mobility, node failure, etc), robustness is an important goal for data delivery schemes. To evaluate the robustness of the proposed RCP protocol, we assume each node has an independent probability of churn, and measure the odds that the shared data can be successfully delivered. The results are plotted in Fig. 8. As shown, the RCP protocol with higher redundancy (i.e., the ratio of number of packets m over data recovery threshold k) can achieve higher delivery rate. For reader's reference, we also display the delivery rate of the naive solution that sends the data along the shortest path. The results justify the advantages of the proposed multipath based data delivery strategy.

VIII. Related Work

In this paper, we utilize linear network coding [4], [5] to achieve both confidentiality and robustness for data delivery in federated coalition networks [1]–[3].

There are some existing work studying the problem of secure network coding [6], [19]–[22]. They introduced a communication system on a wiretap network (CSWN) in which there are eavesdroppers who have access to some of the links in the network. In this context, it is usually assumed that (1) the locations of eavesdroppers, i.e., which link is monitored by the eavesdropper, is unknown (otherwise it would be better to take special protection measures for the monitored links or directly destroy the eavesdroppers). (2) there is an upper bound on the capability of the eavesdroppers, i.e., the number of links can be monitored by them (otherwise there is no security-guaranteed solutions). Given the assumptions, these works either explore the end-to-end throughput of secure traffic given the eavesdropping capacity, or develop probabilistic approaches to minimize the probability that the eavesdroppers can collect enough coded packets to reconstruct the original message. The problem solved in this paper has different system model and setting from these works. In particular, we know which nodes are owned by which parties, and don't assume any bound on the number of nodes of each party. Our work aims at minimizing the cost of computation and communication via a deterministic way that can also ensure the reliability and confidentiality of data delivery.

Another category of strategies [7]–[9], [23]–[25] also bear some resemblance to our work. They make use of secret sharing schemes [26], [27] in combination with multi-path routing [28] to achieve similar goal as secure network coding. The basic idea is to transform a secret message into multiple shares and then deliver them via multiple (independent) paths to the destination. By the secret sharing theory, the secret message cannot be compromised if the number of compromised message shares is smaller than a threshold. Similar to secure network coding schemes, these works usually target on maximizing the security, by either minimizing the probability that enough number of message shares are compromised, or maximizing the overall cost the adversary has to pay to attack the nodes (links). It is clear that these works are different from ours either, in terms of the problem setting and system model.

IX. Conclusion

In this paper, we propose to use linear network coding to achieve robust confidentiality preserving data delivery in federated coalition networks. It circumvents the issues that conventional encryption-based methods need to address due to the unique friendly-but-curious environment of coalition networks, including lack of central authority, expensive key management, frequent node and party churns. It is computationally light weight, and naturally incorporates multipath delivery to ensure robustness against the dynamic topology changes in coalition networks. We formulate the problem of confidentiality preserving and optimal-cost path selection for coded packets as an integer programming problem, relax it into a linear program, and solve it via column generation framework that can address large numbers of variables. Our evaluation demonstrate the effectiveness of our algorithm for large coalition networks consisting of a few thousands nodes.

References

[1] P. R. Smart, D. Mott, E. Gentle, D. Braines, W. Sieck, S. Poltrock, P. Houghton, A. Preece, M. Nixon, M. Strub *et al.*, "Holistan revisited: Demonstrating agent-and knowledge-based capabilities for future coalition military operations," 2008.

[2] S. Calo, D. Wood, P. Zerfos, D. Vyvyan, P. Dantressangle, and G. Bent, "Technologies for federation and interoperation of coalition networks," in *FUSION*, 2009.

[3] Q. Zeng, J. Lobo, P. Liu, S. Calo, and P. Yadav, "Safe query processing for pairwise authorizations in coalition networks," in *ACITA*, 2012.

[4] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, pp. 1204–1216, 2000.

[5] S.-Y. Li, R. W. Yeung, and N. Cai, "Linear network coding," *Information Theory, IEEE Transactions on*, vol. 49, no. 2, pp. 371–381, 2003.

[6] S. Katti, J. Cohen, and D. Katabi, "Information slicing: Anonymity using unreliable overlays," in *NSDI*, 2007.

[7] W. Lou, W. Liu, and Y. Fang, "Spread: Enhancing data confidentiality in mobile ad hoc networks," in *INFOCOM*, 2004.

[8] W. Lou and Y. Kwon, "H-spread: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks," *Vehicular Technology, IEEE Transactions on*, vol. 55, no. 4, pp. 1320–1330, 2006.

[9] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad hoc networks," in *ICNP*, 2001.

[10] Y. R. Yang, X. S. Li, X. B. Zhang, and S. S. Lam, "Reliable group rekeying: Design and performance analysis," in *SIGCOMM*, 2001.

[11] G. Desaulniers, J. Desrosiers, and M. M. Solomon, *Column generation*. Springer-Verlag New York Incorporated, 2005, vol. 5.

[12] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. Savelsbergh, and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations research*, vol. 46, no. 3, pp. 316–329, 1998.

[13] D. Bertsimas and J. N. Tsitsiklis, "Introduction to linear optimization," 1997.

[14] M. E. Lübbecke and J. Desrosiers, "Selected topics in column generation," *Operations Research*, vol. 53, no. 6, pp. 1007–1023, 2005.

[15] W. Lou and Y. Fang, "Predictive caching strategy for on-demand routing protocols in wireless ad hoc networks," *Wireless Networks*, vol. 8, no. 6, pp. 671–679, 2002.

[16] JGraphT, http://jgrapht.org/.

[17] CPLEX, http://www.ibm.com/software/integration/optimization/cplex-optimizer/.

[18] F. Margot, "Symmetry in integer linear programming," in *50 Years of Integer Programming 1958-2008*. Springer, 2010, pp. 647–686.

[19] N. Cai and R. W. Yeung, "Secure network coding," in *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*. IEEE, 2002, p. 323.

[20] T. Chan and A. Grant, "Capacity bounds for secure network coding," in *AusCTW*, 2008.

[21] J. Tan and M. Médard, "Secure network coding with a cost criterion," in *WiOpt*, 2006.

[22] R. W. Yeung and N. Cai, "On the optimality of a construction of secure network codes," in *ISIT*, 2008.

[23] P. Papadimitratos and Z. J. Haas, "Secure data transmission in mobile ad hoc networks," in *WiSe*, 2003.

[24] P. P. Lee, V. Misra, and D. Rubenstein, "Distributed algorithms for secure multipath routing," in *INFOCOM*, 2005.

[25] K. Ren, W. Lou, and Y. Zhang, "Leds: Providing location-aware end-to-end data security in wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 5, pp. 585–598, 2008.

[26] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[27] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *Journal of the ACM (JACM)*, vol. 36, no. 2, pp. 335–348, 1989.

[28] P. Papadimitratos, Z. J. Haas, and E. G. Sirer, "Path set selection in mobile ad hoc networks," in *MobiHoc*, 2002.