LSA: Understanding the Threat of Link-Based Scapegoating Attack in Network Tomography

Xiaojia Xu, Yongcai Wang, *Member, IEEE*, Lanling Xu, and Deying Li

***Abstract*—Network tomography is a powerful and convenient tool to infer the internal states of a network via end-to-end path measurements. However, this article shows that when identifiability is not satisfied at all links (which is general in applications for cost saving purpose), a special kind of attack, i.e., *link-based scapegoating attack (LSA)* that can greatly degrade the network performance while misleading the network tomography to identify some normal links as problematic links can be launched. To fully understand the threat from LSA, this article in particular investigates the conditions and efficient methods to launch LSA from an attacker's point of view. Specifically, an efficient algorithm to figure out the potential scapegoat links is first proposed, and the conditions to launch LSA are presented. Then a *minimum-cost scapegoating attack (MCSA)* problem is proposed, which studies how to manipulate the least-cost link sets to launch LSA. A weighted set cover model and a greedy approximation algorithm are designed to solve *MCSA* problem, which is approached by a linear programming method with H_K approximation ratio. Evaluation on both synthetic and real network topologies shows the wide existence of LSA threats, the feasibility of the proposed LSA conditions, and the attacking strategies.**

***Index Terms*—Network tomography, scapegoating attack, minimum cost, vulnerable link set, unobserved cut set.**

I. Introduction

NOWADAYS, networked equipment and systems are becoming increasingly pervasive and complex. Accurately and timely knowing the internal states of the network, such as the link delay, packet loss rate, and jitter is an essential requirement in network management. The traditional network management protocols (such as special diagnostic tools [1], traceroute [2], flow monitoring [3], etc.) become difficult to access the internal components to obtain node or link status or to find potential anomalies or failures.

Network tomography [4], using end-to-end path measurements to infer the internal states of the networks instead of measuring the elements within the networks directly, becomes a promising solution [5], [6], [7], [8]. Network tomography only measures the end-to-end path performances between a selected set of monitors. By solving state recovering functions, network tomography infers the internal states of nodes and links [9], [10], [11], [12], [13]. It avoids the issues such as high measurement cost and high internal measurement overhead of the direct measurement methods.

The "identifiability" problem is a critical problem in network tomography [9], which indicates whether the internal states of the networks can be uniquely recovered by the end-to-end external path measurements. For example, to recover additive link metrics, such as delays or logarithmic form loss rates, the condition of unique state recovery is that *the rank of the routing matrix equals the number of links* [9]. However, due to the large number of links, unless the number of probing paths is not less than the number of links, this identifiability condition is hard to be satisfied, which requires very high measurement costs.

Approaches have been investigated on the feasibility and algorithms for recovering the internal states correctly for rank deficient routing matrix. These methods can be roughly classified into four categories:

1) Investigation of the feasibility and conditions for state recovery for different kinds of networks [8], [14], [15].
2) Design of the recovering algorithms [9], [16].
3) Routing matrix design for improving identification performance using limited probing budget [10], [17].
4) Monitor deployment optimization for improving identification using limited monitoring cost [6], [18], [19]. A common agreement of existing studies is that for triconnected components, the monitors and probing paths can be designed appropriately to guarantee the identifiability of all the links in the components [20]. Whereas, for non-triconnected components, identification is still hard to be guaranteed.

When identification is not guaranteed, a new type of attack, i.e., *scapegoating attack*, becomes a critical threat to the network tomography. Chiu et al. [21] considered degrading the performance of the networks by injecting delays to several path measurements, and the network tomography cannot precisely localize the attackers. Zhao et al. [22] pre-selected scapegoat links and studied how to maximize the attacking damage or obfuscation by manipulating path measurements. They further investigated how to launch the attack in [23] by manipulating path measurements and making some normal nodes as scapegoats. However, these existing methods hack the path measurements, which is hard to accomplish because the path measurements are generated by the network tomography system. A more practical way is to hack some link metrics to trigger the scapegoating attack.

Manuscript received 13 January 2023; revised 7 March 2023; accepted 24 April 2023. Date of publication 27 April 2023; date of current version 25 October 2023. This work was supported by the National Natural Science Foundation of China under Grants 61972404 and 12071478. Recommended for acceptance by Dr. Bin Xiao. *(Corresponding author: Yongcai Wang.)*

The authors are with the School of Information, Renmin University of China, Beijing 100872, China (e-mail: xuxiaojia@ruc.edu.cn; ycw@ruc.edu.cn; 2018202168@ruc.edu.cn; deyingli@ruc.edu.cn).

Digital Object Identifier 10.1109/TNSE.2023.3271135

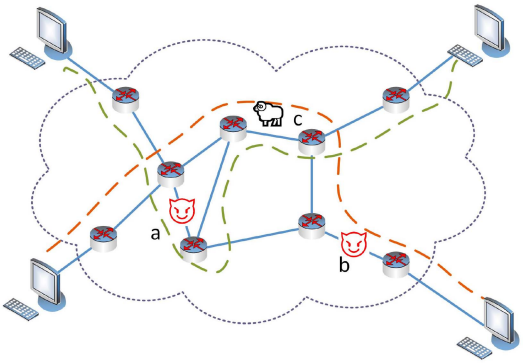

Fig. 1. An example of the scapegoating attack, in which a and b are the manipulated links but c is identified as problematic link mistakenly.

This article considers the *link-based scapegoating attack (LSA)*, which is to manipulate some link metrics to degrade the network performances while misleading network tomography to detect some normal links as problematic so that they become scapegoats of the hacked links. The packet loss attacks include black hole attack [24] and grey hole attack [25] can be utilized to carry out LSA by dropping or delaying packets on specific links. Fig. 1 shows an example of LSA. There are four monitors in the network, which are represented by the computers. Suppose network tomography measures the end-to-end path delays of the two probing paths (shown by the dashed lines) and wants to recover the delays of the traversed links. Obviously, the link states cannot be uniquely recovered because the routing matrix is not full rank. In such a case, when an attacker hacks the two links a and b to inject large delays, the network tomography algorithms such as pseudo-inverse [21], [22] tend to mistakenly identify the link c as the problematic link because if c has the large delay it will cause the same phenomenon observed by the measurements. A reasonable algorithm generally would like to identify the minimal number of problematic links that can cause the observed phenomenon. This assumption holds when the attacker wants to minimize the cost of attack. As in this example, c is identified abnormal, which is a scapegoat. To fully understand the threat of LSA, this article investigates the conditions and efficient methods to launch LSA from an attacker's point of view. In particular, the following key problems are studied.

(1) Given a network that does not satisfy the identifiability property, how to efficiently figure out which set of links are potential scapegoats? Note that it may not know the path measurements from an attacker's point of view. (2) What is the necessary and sufficient condition to successfully launch LSA when there are potential scapegoat links? (3) It generally incurs some cost to attack some specific link. How to manipulate the minimum cost link set to launch LSA successfully? The detailed contributions of our paper are as follows:

1) An efficient scapegoat link detection (SLD) algorithm is proposed to locate the potential scapegoat link set by a proposed Gaussian Elimination method in non-triconnected components. The algorithm is efficient even in large networks.
2) A necessary condition and a sufficient condition to successfully launch LSA in network tomography and to mislead the network tomography to identify some normal links as scapegoats are presented.
3) Then a *minimum cost scapegoating attack (MCSA)* problem is proposed, which studies how to manipulate the minimum cost link set to launch LSA successfully. A set cover model is proposed, and an approximation greedy algorithm with H_K ratio is developed to solve MCSA.
4) Extensive verifications on both real ISP network topologies and synthetic topologies show the wide existence of LSA threats, the effectiveness of the proposed LSA conditions and the attacking strategies.

The remainder of our paper is organized as follows. Section II analyzes the related work. Section III formulates our problem. Section IV designs the scapegoat link detection algorithm to locate the vulnerable link set. Section V theoretically analyzes the necessary conditions to launch scapegoating attack successfully and proposes the minimum cost scapegoating attack problem. Section VI discusses the evaluations of proposed algorithms. Finally, Section VII concludes the paper.

II. Related Work

A. Network Tomography and the Rank Deficiency Problem

Network tomography calculates the performance of links in a network through end-to-end path measurements [4] and has been used for inferring link states and conducting fault locations in wired networks [5], and wireless networks [26]. Early work has focused on finding the most likely network state from the given measured path values. Shih et al. [27] studied the estimation of link delay distribution from unicast edge measurements. Based on end-to-end multicast and unicast observations, Adams et al. [28] inferred the internal topological structure. Xi et al. [29] introduced the detection mode of dual broadcasting. Duffield et al. [7] proposed using the back-to-back unicast flow as the probing paths to infer the packet loss rate of the links.

For additive link metrics, uniquely identifying all link metrics requires the rank of the routing matrix to be equal to the number of links, which is, however, difficult to satisfy due to the large number of links [10]. Rank deficiency of the routing matrix will cause ambiguity in the solution. Chen et al. [8] observed that an arbitrary set of measurements is usually insufficient to identify all the link states and proposed the General Method of Moments to estimate the link state distribution. Nguyen et al. [16] proposed to address the ambiguity problem by using a small set of measurements using Boolean algebra to learn priors. Tati et al. [9] investigated the problem to maximize the expected rank under a budget constraint on the probing cost. They proved that the problem is NP-hard and proposed a RoMe algorithm with a guaranteed approximation ratio.

B. The Efforts to Guarantee Identifiable

Other studies have investigated the design of the probing paths and the placement of monitors to ensure identifiability. Xia et al. [14] proved that if a network is directed unless every non-isolated node is a monitor, link metrics can not be all identifiable.

When the network is undirected, Gopalan et al. [15] derived the first sufficient and necessary conditions for identifying all link metrics when the probing paths may contain cycles.

For cycle-free probing paths, Ma et al. [10] proved that the link metrics are identifiable by placing k monitors appropriately if the extended graph $\mathcal{G}_{ex}$ is 3-vertex connected, where $\mathcal{G}_{ex}$ is obtained by adding two virtual monitors m_1', m_2' and $2k$ virtual links between each pair of virtual-actual monitors into $\mathcal{G}$. This condition indicates that the identifiability property depends on the network topology. Ma et al. [6] further studied how to optimize the placement of a limited number of monitors to maximize the number of identifiable links. He et al. [18] proposed the monitor placement optimization problem in dynamic networks. Dong et al. [30] proposed an optimal monitor assignment algorithm for preferential link tomography. In sparse networks, guaranteeing identifiability needs to select a large number of monitors. On the other hand, given a set of monitors, Ma et al. [11] and Pepe et al. [31] studied the optimization of the probing paths. To satisfy the sufficient and necessary conditions for identifying all link metrics, the high costs of monitor placement and path selection are unavoidable. However, network tomography is designed for avoiding high internal measurement overhead of the direct measurement methods. Identifiability is generally hard to be achieved with limited monitoring and probing budget.

C. Being Not Identifiable Leaves Risks

When identification is not guaranteed, methods like WCF estimator [8], Pseudo-inverse and congested link identification algorithm [16] are studied to recover the link states. Recently, the risk of being attacked has also been noted. Chiu et al. [21] proposed the *scapegoat attacking (SA)* problem. They considered degrading the performance of the networks by injecting delays into several path measurements, and network tomography cannot localize the attackers. Zhao et al. [22] studied chosen-victim, maximum-damage, and obfuscation scapegoating attacks when end-to-end path measurements are maliciously manipulated. Zhao et al. [23] further investigated SA when some path measurements are manipulated, and some nodes are treated as scapegoats. However, these studies considered to hack the path measurements directly, which is hard be realized in network tomography, because the path measurements are taken by the network tomography administrator. This article studies a more realistic problem, where the attacker attacks some links in the network. This can be carried out by simply injecting delay or dropping packets at specific links [24], [25]. The attacker doesn't need to access, nor need to know the path measurement results of the network tomography manager. Feng et al. [32] studied the bound-based network tomography, where the bounds are derived for unidentifiable links. But if the scapegoating attack is launched in the bound-based network tomography, some unidentifiable normal links may still be attacked as scapegoats. Their link performances maybe beyond the bounds and so they are detected as abnormal links, which become the scapegoats."

Instead of hacking the path measurements, link-based scapegoating attack (LSA) can be initialized by the packet loss attack, which destroys the network by dropping or delaying packets. Packet dropping attacks mainly include black hole attack [24] and grey hole attack [25]. The black hole attack attracts and discards all packets routed to a malicious node, whereas a grey hole attack is a selective forwarding attack that only discards selected packets. For the existence of these link-based attacking methods, the LSA problem must be noted for unidentifiable network tomography systems. But to the best of our knowledge, the threats of LSA are not well understood in the literature yet. A thorough investigation of LSA is presented from an attacker's point of view in this article.

TABLE I
MAIN NOTATIONS

Symbol	Definition
$V(\mathcal{G})$	the set of nodes in graph $\mathcal{G}$
$L(\mathcal{G})$	the set of links in graph $\mathcal{G}$
m_i	the i-th monitor in $\mathcal{G}$, $m_i \in V(\mathcal{G})$
Y	the set of measured path metrics
$\widehat{Y}$	the set of pseudo path metrics
x_i	the link metrics of link l_i in real world
$\widehat{x}$	the estimated link metrics from network tomography
Δx_i	the injected delays to the manipulated link l_i
β_{min}	the threshold for being a normal link
β_{max}	the threshold for being identified as a problematic link
L_m	the set of manipulated links
L_s	the set of scapegoat links
$\mathcal{B}_i$	the i-th biconnected component in $\mathcal{G}$
$\mathcal{T}_i$	the i-th triconnected component in $\mathcal{G}$
P_s	the scapegoat path set
P_m	the manipulated path set
Γ	the threshold used by network tomography to diagnose path states
C	the cut set of a path set P

III. Problem Formulation

A. Network Tomography Model

We model a network as a weighted, undirected graph $\mathcal{G} = (V, L, X)$, where V is the set of nodes and L is the set of links, $|V| = n$ and $|L| = m$. Set X represents the link metrics, where $x_i \in X$ is link i's performance metric to be estimated, such as latency, jitter, and loss rate. We consider the link metrics are additive, which is a canonical model for representing performance measures [10], [11], [19], [21], [22]. The delay is directly additive, and the packet loss rate or packet transfer rate is also additive under logarithmic form. Table I summarizes the main notations used in our paper.

$M = \{m_i\}$ is the set of monitors, a subset of nodes used for injecting and extracting probing packets. A probing path p_i for network tomography denotes a sequence of links that starts from a source monitor s_i and ends at a destination monitor d_i. y_i is the end-to-end measurement of this probing path p_i. $P = \{p_i\}$ is the set of paths and $Y = \{y_i\}$ is defined as the set of path measurements. The routing matrix $R = \{r_{i,j}\}_{p_i \in P, l_j \in L}$ models how the links in L is covered by the probing paths. $r_{i,j} = 1$ if a link $l_j \in L$ is on the path $p_i \in P$ and $r_{i,j} = 0$ otherwise.

Network tomography is to find a solution $\hat{x}$ that represents the estimated link metrics by solving equation $R\hat{x} = Y$. When the routing matrix R has full column rank, i.e., $\text{Rank}(R) = m$, this linear equation has a unique solution, called *identification condition*. However, since the number of links is much larger than the number of nodes, i.e., $m \gg n$, it is difficult to generate probing paths satisfying the identification condition. Users, therefore, generally adopt the routing matrix having $\text{Rank}(R) < m$. In such a case, Pseudo-inverse is used [21], [22] to estimate $\hat{x}$ by:

$$\hat{x} = (R^T R)^{-1} R^T Y \tag{1}$$

Link states are diagnosed by the recovered link metrics. Without loss of generality, we assume the link metric is the link delay. Then state diagnosis can be simplified as follows:

Definition 1 (Estimated Link States): Given estimated link delay $\hat{x}$, a link may be classified into three states:

$$\psi(l_i) = \begin{cases} normal & \text{if } \hat{x}_i < \beta_{\min} \\ uncertain & \text{if } \beta_{\min} \leqslant \hat{x}_i \leqslant \beta_{\max} \\ problematic & \text{if } \hat{x}_i > \beta_{\max} \end{cases} \tag{2}$$

where $\beta_{\min} < \beta_{\max}$ are two delay thresholds. When the estimated link delay is below $\beta_{\min}$, the link will be thought "normal". When $\hat{x} > \beta_{\max}$, the link will be detected as "problematic" and "uncertain" if in between.

B. Attacking Model

When identifiability is not guaranteed, the link state diagnosis models in (1), (2) leave room for malicious attacks. Chiu et al. [21] showed that by injecting delays into the path measurements, the recovered delays of some links could be greatly increased. Chiu et al. [21] injected delay vector z into path measurement Y. So the recovered link states are

$$\hat{x} = (R^T R)^{-1} R^T (Y + z) \tag{3}$$

So the attacks are injected on the path measurements Y, which can greatly impact the recovered $\hat{x}$. However, it is difficult for attackers to hack the path measurement vector Y directly because the measurements belong to the network tomography system. On the other hand, Chiu et al. [21] did not consider attributing the total degradation to particular links and required the manipulated links to inject different delays in different paths, which is hard to be realized.

This article considers a more practical way to inject delays to a set of manipulated links while making the network tomography detect a set of normal links as scapegoats. Let L_s denotes the scapegoat link set. Let L_m denote the manipulated link set. The conditions on how to select a set of links L_m to inject delays to make L_s be scapegoat while the attacks on L_m cannot be discovered are studied. How to select the minimum cost L_m set to lunch LSA, which is called the MLSA problem is then proposed. Note that $L_s \cap L_m = \emptyset$ should be satisfied since the manipulated links should not be detected as problematic.

Fig. 2 summarizes the procedure of launching the scapegoating attack under the watch of network tomography. Note that this article stands on the attacker's point of view. The attacker knows only $\mathcal{G}$ and R and doesn't know the path measurements Y.

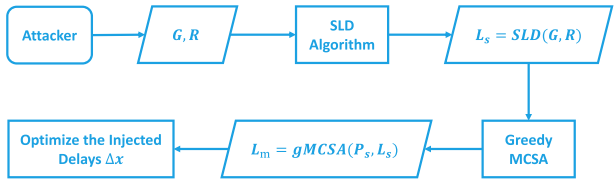

Fig. 2. The procedure of launching the scapegoating attack.

After selecting L_m, the attacker will consider how to optimize injected delays Δ_x for the links in L_m, where $\Delta x_i > 0$ for $l_i \in L_m$ and $\Delta x_i = 0$ for $l_i \notin L_m$, such that:

$$R(x + \Delta x) = Y' \tag{4}$$

So the LSA problem on the attacker side is how to select L_m and how to determine Δx_i for each link in L_m, which is different from traditional path-based scapegoating attack. Note that Y' is the naturally obtained measurements by the network tomography administrator after the attackers manipulated L_m. The link states are then recovered by the network tomography administrator as:

$$\hat{x} = \left(R^T R\right)^{-1} R^T Y' \tag{5}$$

Using the rank deficient of the routing matrix, the attacker hopes that in the recovered link states, there are:

$$\begin{cases} \hat{x}_i \leq \beta_{\min} \text{ for } l_i \in L_m \\ \hat{x}_i > \beta_{\max} \text{ for some } l_i \in L_s \end{cases} \tag{6}$$

So that the attacks from L_m mislead the network tomography to identify the scapegoats in L_s as problematic links mistakenly. If (5), (6) hold for L_m and some links in L_s, we call such a LSA is successfully launched. Network tomography also has some basic rules to detect *problematic path measurements*. When the measured delay of a path is less than a threshold Γ, the path measurement is thought "normal". Otherwise, it is thought "problematic". To avoid being discovered, the attacker should also ensure all the affected paths are "normal", so that the network tomography can accept the affected path measurements to calculate the link delays by (5).

IV. Algorithm to Locate the Potential Scapegoat Links

Given $\mathcal{G}(V, L), R$, the first problem is how to locate which set of links are potential scapegoats. The algorithm proposed in this section does not need to access the real measurement Y. The attacker only needs to assume a set of pseudo path metrics $\widehat{Y}$ based on the structure of $\mathcal{G}$ and R. The identifiability of links can be evaluated by two levels: (1) network topology level without considering specific probing paths and (2) recovery level regarding specific probing paths. The attacker assumes that the network administrators running network tomography are qualified. So the attacker aims to find links whose identifiability cannot be guaranteed. We firstly introduce some results from existing work [20].

Theorem 1: In a triconnected component $\mathcal{T}_i$ of a graph $\mathcal{G}$, all of its link metrics are identifiable if we assign any three vertices in

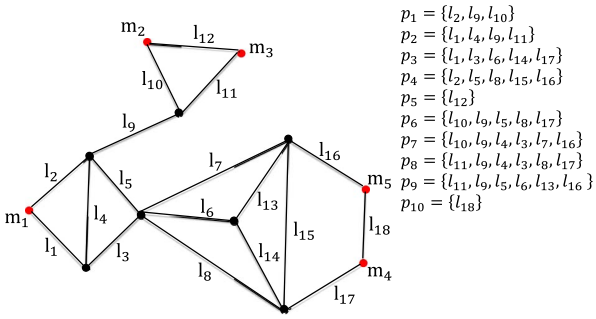

Fig. 3. A network with five monitors selected by MMP [20] algorithm. Ten end-to-end paths are listed in the right part to identify the link metrics.

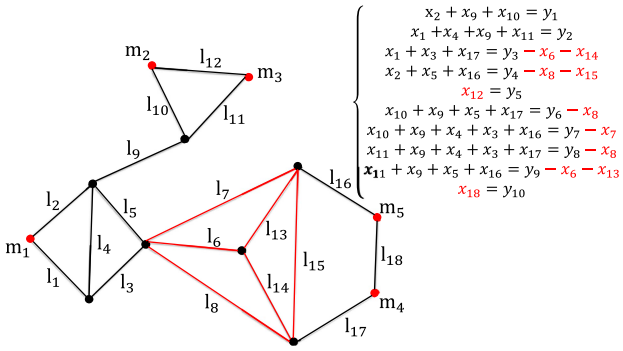

Fig. 4. The left part is the highlight of links in triconnected components. The right part is the transformation of linear system.

the original graph $\mathcal{G}$ as monitors and the three vertices v_1, v_2, v_3 satisfy one of the following conditions [20]:

1) v_1, v_2, v_3 are in $\mathcal{T}_i$;
2) one or more vertices $v_j, j \in \{1, 2, 3\}$ is not in $\mathcal{T}_i$, but there are distinct probing paths (without any repeated vertex) from v_j to $\mathcal{T}_i$.

A. Detect the Triconnected Components

According to Theorem 1, if a triconnected component $\mathcal{T}_i$ has at least three eligible vertices as its monitors, all links in the $\mathcal{T}_i$ are identifiable. Considering the network administrator is a qualified defender, the attacker should not attack the links in the triconnected components. This helps to narrow down the search space of the scapegoat link set after knowing the network topology. Algorithms, such as SPQR-tree [33] can detect triconnected components in a network in linear time. The links in triconnected components can then be excluded from consideration of being scapegoats. However, since the link metrics of the links in triconnected components are unknown, locating the vulnerable link set outside the triconnected components remains a problem. Note that after the set of monitors and the set of probing paths have been known, there may remain a set of links which are not identifiable. We call such link set "vulnerable link set", because the attacker can attack these links without being directly identified.

B. Locate the Vulnerable Link Set

Although the link metrics in the triconnected components are unknown, because they are identifiable, they can be regarded as constant values to infer the identifiability of other links. Note that the single link between two monitors is also identifiable, and they can also be treated as constants. Then monitoring equations containing these links can be simplified by converting the identifiable variables to constants. By denoting the constant values of links by y'_j and through a transformation of the linear system, a reduced dimension linear system can be obtained. An example is presented to show the detailed routine.

Fig. 3 presents a sample network with twelve vertices, eighteen links (l_1 to l_{18}), and five monitors (m_1 to m_5) selected by the monitor placement algorithm [20]. Ten end-to-end paths (p_1 to p_{10}) are constructed from all pairs of monitors to identify the link metrics, as listed in the right part of Fig. 3.

We set x_i as the link metric of l_i and y_j as the estimated path metric of p_j. Then we have the following linear system of equations:

$$\begin{cases} x_2 + x_9 + x_{10} = y_1 \\ x_1 + x_4 + x_9 + x_{11} = y_2 \\ x_1 + x_3 + x_6 + x_{14} + x_{17} = y_3 \\ x_2 + x_5 + x_8 + x_{15} + x_{16} = y_4 \\ x_{12} = y_5 \\ x_{10} + x_9 + x_5 + x_8 + x_{17} = y_6 \\ x_{10} + x_9 + x_4 + x_3 + x_7 + x_{16} = y_7 \\ x_{11} + x_9 + x_4 + x_3 + x_8 + x_{17} = y_8 \\ x_{11} + x_9 + x_5 + x_6 + x_{13} + x_{16} = y_9 \\ x_{18} = y_{10} \end{cases} \tag{7}$$

By using the algorithms that partition graphs into biconnected components and then into triconnected components, we can know that the red links in Fig. 4, i.e., $\{l_6, l_7, l_8, l_{13}, l_{14}, l_{15}\}$ are all links belonging to triconnected components. They are firstly excluded from L_s. Note that these constants are moved to the right side of (8):

$$\begin{cases} x_2 + x_9 + x_{10} = y'_1 \\ x_1 + x_4 + x_9 + x_{11} = y'_2 \\ x_1 + x_3 + x_{17} = y'_3 \\ x_2 + x_5 + x_{16} = y'_4 \\ x_{10} + x_9 + x_5 + x_{17} = y'_5 \\ x_{10} + x_9 + x_4 + x_3 + x_{16} = y'_6 \\ x_{11} + x_9 + x_4 + x_3 + x_{17} = y'_7 \\ x_{11} + x_9 + x_5 + x_{16} = y'_8 \end{cases} \tag{8}$$

By Gaussian Elimination of the above linear system, we can derive that $x_1 = \frac{y'_2 + y'_3 - y'_7}{2}$, $x_2 = \frac{2y'_1 + 2y'_4 - y'_5 - y'_6 + y'_7 - y'_8}{4}$ and $x_4 = \frac{2y'_4 - 2y'_1 + 2y'_2 - 2y'_3 + y'_5 + y'_6 + y'_7 - 3y'_8}{4}$. Therefore, l_1, l_2 and l_4 are identifiable and they should not be put into L_s.

The remained links are not identifiable, so we put them into candidate set of scapegoat link, i.e., $L_s = \{l_3, l_5, l_9, l_{10}, l_{11}, l_{16}, l_{17}\}$. Links in L_s are potential scapegoats. Note that both the tri-connected component partition and the Gaussian elimination steps are highly efficient even for large scale networks.

Algorithm 1: Scapegoat Link Detection Algorithm $L_s = \text{SLD}(\mathcal{G}, R)$.

Input: A network graph $\mathcal{G}(V, L)$, the routing matrix of the network tomography R
Output: L_s

```
1  Initialize L_s = ∅, Ŷ = ∅, L = L(G);
2  for each path p_i ∈ P do
3      for each link l_j ∈ L do
4          ŷ_i = ŷ_i + R(i, j);
5  partition G into biconnected components B_1,B_2,...;
6  for each biconnected component B_i do
7      partition B_i into triconnected components
       T_1,T_2,...;
8  for each triconnected component T_j do
9      for each link l_k ∈ L do
10         if l_k ∈ T_j then
11             L = L \ l_k;
12             for each path p_e ∈ P do
13                 ŷ_e = ŷ_e − R(e, k);
14                 R(e, k) = 0;
15 [R' Ŷ'] = GaussianElimination[R Ŷ];
16 for each link l_k ∈ L do
17     if l_k has the unique solution in R' then
18         L = L \ l_k;
19 L_s = L;
20 return L_s;
```

Algorithm 1 gives SLD algorithm for potential scapegoat links. The algorithm firstly initializes the candidate set L_s, the set of pseudo path metrics $\widehat{Y}$, and puts all links of the graph $\mathcal{G}$ into set L (line 1). Then without knowing the path measurements, the attacker simply sets the values of the pseudo path metrics $\widehat{y}_i$ as the total number of links on the p_i, $\widehat{y}_i = \sum_{j=1}^{m} \mathcal{R}(i, j)$ (lines 2 to 4). The setting of pseudo path metrics $\widehat{Y}$ can ensure the efficiency and accuracy of judging the identifiability of links. The algorithm then searches all biconnected components to find all triconnected components (lines 5 to 7). The links in triconnected components are excluded from the candidate set of scapegoat links, and they are treated as constants (lines 8 to 14). Then Gaussian Elimination is applied to the dimension reduced linear system, and all links that have the unique solution in R' are further excluded from the candidate set of scapegoat links. Then the final candidate set of scapegoat links (lines 15 to 20) is obtained. Partitioning a connected graph into biconnected components and triconnected components can both be done in linear time [33]. The time complexity of Gaussian Elimination (line 15) is $O(m^3)$, where m is the number of links in $L(\mathcal{G})$. Therefore, the time complexity of SLD algorithm is $O(tmk + m^3)$, where t is the number of triconnected components and k is the number of paths in P. By SLD algorithm, the attacker can quickly figure out which set of links are potential scapegoats.

V. Necessary Conditions to Launch Scapegoating Attack

Based on the detected potential scapegoat set L_s, this section studies how to launch the scapegoating attack.

The attacker aims to choose a set of manipulated links L_m from $L \setminus L_s$ so that it can inject delays into L_m while network tomography cannot locate the manipulated links and will mistakenly identify the links in L_s as scapegoats. In practice, there is some cost to manipulate a link, which is related to the difficulty of controlling a link, such as gaining backdoor access to an associated device [34]. Let c_i denote the cost to manipulate a link $l_i \in L$. Then the total cost to manipulate the links in L_m to inject attack is $Cost(L_m) = \sum_{l_i \in L_m} c_i$. Then based on the relationship to L_s and L_m, the probing paths can be divided into three categories.

Definition 2 (Three Types of Probing Paths): The probing paths in P can be divided as:

- $P_m \subseteq P$ denotes the **manipulated path set**. Each path in it passes through at least one manipulated link.
- $P_n \subseteq P$ is the **normal path set**, which contains all measurement paths that haven't passed through any manipulated link.
- $P_s \subseteq P$ denotes the **scapegoat path set**. Each path in it passes through at least one scapegoat link.

Suppose the original link metric of a manipulated link is x_i, and the injected delay is Δx_i. Δt_i denotes the increased delay on a path $p_i \in P_m$, which is the sum of the injected delays from the manipulated links on this path. Since the injected delays only affect the manipulated paths, the following Lemma 1 is easily obtained.

Lemma 1: Given P, L_s, L_m, if delays are injected into links in L_m, there must be $\Delta t_i \geqslant 0$ for $p_i \in P_m$ and $\Delta t_i = 0$ for $p_i \in P_n$.

A. Necessary and Sufficient Conditions to Select L_m

When given L_s and P, how should we choose the manipulated link set L_m to launch the scapegoating attack successfully?

One principle is to choose the manipulated links as little as possible, not only to minimize the attacking cost but also to reduce the risk of being detected. But we still need to make sure the manipulated links satisfy conditions in (5), (6) and can successfully launch the attacks. The following properties can be used to narrow down the search of the manipulated link set. At first, Given P_s, links not on P_s should not be selected as the manipulated links.

Lemma 2: If a path $p_i \notin P_s$, then the injected delay in any link $l \in p_i$ cannot be attributed to any link in L_s by the network tomography model in (1).

The Lemma is straightforward. To attribute the increased delay to the scapegoats, L_m should be selected only on the paths that pass through links in L_s. Meanwhile, to make $l_i \in L_s$ a scapegoat, we need to inject delays to a link set that can cut all the measurement paths that pass through l_i.

Definition 3 (Cut Set of Paths): A cut set C of a path set P denotes a set of links in P. For every path $p_i \in P$, the path p_i

passes through at least one link in C. In other words, if we cut all the links in the cut set C, we would cut all the paths in P.

Definition 4 (Minimal Cut Set): If C is a cut set of P, which cannot be reduced without losing its status as a cut set, then C is a minimal cut set.

From Lemma 2, if a link l in a minimum cut set C is on a path $p \in P_n$ that doesn't go through L_s, then attacks injected on l cannot be attributed to any link in L_s. But the attack will affect the performance of $p \in P_n$. We say such a cut set C is *observed* by the path p.

Definition 5 (Unobserved Cut Set): An unobserved cut set of L_s denotes a set of links. These links cut all the paths that pass through L_s, and there is no other path in P_n passing through any link in this set.

Theorem 2 (Necessary condition for launching the LSA): Given $\mathcal{G}$, L_s, the necessary condition for making l_i be a scapegoat is that the manipulated link set L_m cuts the measurement paths that pass through l_i, and L_m is an unobserved cut set of l_i.

Proof: We prove by contradiction. Let's denote the path set that pass through l_i by P_s^i. We assume that the chosen L'_m is not a cut set for P_s^i. $\exists p'_i \in P_s^i$, all links on p'_i are not manipulated links, so $p'_i \in P_n$. We know from Lemma 1 that: when path $p'_i \in P_n$, $\Delta t'_i = 0$. That means $\forall l'_j \in p'_i$, $\hat{x}_{l'_j} < \beta_{\min}$, So l_i must be normal. This is contradict with the requirement that $\hat{x}_{l_i}$ is problematic. So L_m must be an unobserved cut set of P_s^i for making l_i a scapegoat. □

From the definition of scapegoating attack in Section III, the sufficient condition of successfully launching scapegoating attack by manipulating L_m can also be given.

Theorem 3 (Sufficient condition for successfully launching the LSA): If the following conditions are met, the scapegoating attack can be launched successfully.

$$\begin{cases} R(x+\Delta x) = Y' \\ \hat{x} = \left(R^T R\right)^{-1} R^T Y' \\ \sum_{l_j \in p_i} \Delta x_j = 0, \forall p_i \in P_n \\ \sum_{l_j \in p_i} \Delta x_j \geq 0, \forall p_i \in P_m \\ \sum_{l_j \in p_i} (x_j + \Delta x_j) \leqslant \Gamma, \forall p_i \in P \\ \hat{x}_j \leqslant \beta_{\min}, \forall l_j \in L_m \\ \hat{x}_j > \beta_{\max}, \exists l_j \in L_s \\ \Delta x_j = 0, \forall l_j \in L_n \cup L_s \\ \Delta x_j \geqslant 0, \forall l_j \in L_m \end{cases} \tag{9}$$

Proof: The first two constraints mean the states are recovered by network tomography after the attacker injected delays. The third and fourth conditions restrict the injected delays affect only the manipulated paths. The fifth constraint restricts the injected delays from making the path delay beyond the threshold. The sixth and seventh constraints restrict the recovered states of the attacking and the scapegoat links. The last two constraints indicate the range of variables. So if the conditions in (9) are met, the network tomography identifies some scapegoats as problematic, and the manipulated links have successfully injected attacks and have not been discovered. So the conditions are sufficient for successfully launching a scapegoating attack. □

So the routine of attack is to firstly select L_m and then optimize the injected delays Δx for L_m, which will be detailed in the following sections.

B. Optimize the Selection of L_m

Since there are multiple combination unobserved cut sets that can cut P_s, L_m can be selected with some optimization purpose. If all links are assigned unified cost, finding the minimum cut set resembles finding the cut set with the least number of links. This article considers a general form of manipulating cost, i.e., $c_i \geq 0$. It can be instanced as the cost to conduct grey hole attack [25] to cause packet dropping at a specific link. The minimum cost scapegoating attack problem is then considered. Among all the unobserved cut sets of the path set P_s, the one with the minimum sum attacking cost is called the minimum cost unobserved cut set of P_s.

Definition 6 (Minimum Cost Scapegoating Attack (MCSA) Problem): Given $\mathcal{G}$, L_s, P and link cost $\mathcal{C}$, the minimum cost link-based scapegoating attack problem is to find the minimum cost link set L_m to inject proper delays Δx for links in L_m, such that the conditions in (5), (6) can be satisfied.

The problem reduces into two subproblems:

1) Find the minimum cost unobserved cut set and use the links in the cut set as L_m;
2) Optimize the injected delays $\{\Delta x_i\}$ for links $l_i \in L_m$ to satisfy the conditions in (5), (6).

C. Solving the MCSA Problem

Lemma 3: A minimum cost cut set must be a minimal cut set.

Proof: If the minimum cost cut C is not a minimal cut set, C can be reduced by removing a link l and remaining as a cut. Since the cost of the link l is positive, the reduced cut set $C \setminus l$ has a lower cost than C, which contradicts with that C is the minimum cost cut set. □

Theorem 4 (NP-Hardness): Given $\mathcal{G}, P, L_s$, and $\mathcal{C}$, finding the minimum cost unobserved cut set $C_{\min}$ of L_s is NP-Hard.

Proof: The problem of finding the minimum cost unobserved cut set can be proved by a reduction from Weighted Set Cover Problem (WSCP). We know the weighted set cover problem is NP-hard. In the weighted set cover problem, we are given a universe of n elements $U = \{e_1, e_2, \ldots, e_n\}$ and a family of m subsets of U, $F = \{S_1, S_2, \ldots, S_m\}$. Each subset has a cost $c(S_i) \in \mathbb{R}^+$. Our goal is finding the minimum cost subset $F' \subseteq F$, such that $\bigcup_{S_j \in F'} S_j = U$.

The WSCP problem can be reduced to the following instance of minimum cost unobserved cut set problem. $P_s = \{p_1, p_2, \ldots, p_n\}$ in one-to-one correspondence with the set of elements $U = \{e_1, e_2, \ldots, e_n\}$. Let $L = \{l_1, l_2, \ldots, l_m\}$ be all the links on P_s but not on any P_n. It is one-one correspondence with the collection of subsets $F = \{S_1, S_2, \ldots, S_m\}$. The relationship between the links and the paths is such that link l_i is traversed by path p_j if and only if set s_i covers element e_j. Let the link attacking cost c_i be one-one correspondence to the set cost $c(S_i)$. Then, there is a minimum weight cover F^* for U, if and only if there is a minimum cost set C^* that cuts P_s and is not observed by any other path in P_n.

Algorithm 2: Greedy MCSA $C = \text{gMCSA}(P_s, L_s)$.

Input: P_s, L_s
Output: C
Initialize $C = \emptyset$, $P = P_s$;
while $P \neq \emptyset$ **do**
  Choose $l \in P \setminus (L_s \cup L_o)$ that minimizes $\frac{c_l}{|P_s(C \cup l)| - |P_s(C)|}$;
  $C = C \cup l$;
  $P = P \setminus (P_s(C \cup l) - P_s(C))$;

Suppose there is a minimum weight set cover F^*, such that $\bigcup_{S_j \in F^*} S_j = U$. It corresponds to a link set $C^* \subseteq L$ that cuts P_s and is not traversed by P_n from the correspondence. The sum weight of F^* is equal to the sum cost of C^*. If C^* is not the minimum cost cut set, there must be a cut set with a lower cost. This will correspond to a set with a lower weight to cover U, which is contradict that F^* is the minimum weight set cover. So C^* must be the minimum cost cut set. Similarly, if C^* is a minimum cost cut set, it cuts every path of P_s. So F^* is a set cover of U. We can also prove similarly by contradiction that F^* is the lowest weight set cover. So the MCSA is NP-hard since the WSCP problem is NP-hard. □

D. Greedy Algorithm to Approach MCSA

A greedy algorithm is proposed to approach MCSA. Let L_o be the set of links in P_s which are also observed by paths in P_n. The greedy algorithm finds a cut set C by repeatedly selecting a link l in $P_s \setminus (L_s \cup L_o)$ that minimizes the cost c_l divided by the number of paths in P_s not yet cut by C. It stops and returns C when all paths in P_s are cut.

In Algorithm 2, $|P_s(C)|$ indicates the number of paths in P_s cut by C. So $\frac{c_l}{|P_s(C \cup l)| - |P_s(C)|}$ evaluates the cost over contribution of l. The algorithm greedily selects the link with the lowest cost-contribution ratio. $P_s(C \cup l) - P_s(C)$ is the newly cut paths by l.

Using the greedy MCSA algorithm, we can find the approximate solution in polynomial time and the time complexity of gWCSA is $O(mk)$, where m is the number of links in $P_s \setminus (L_s \cup L_o)$ and k is the number of paths in P_s.

Lemma 4 (Approximation ratio of gMCSA): Let K be the largest traversal of links in $P_s \setminus (L_s \cup L_o)$ on P_s. Let $H_K = \sum_{i=1}^{K} 1/i \approx \ln K$, then the gMCSA algorithm returns a cut set of cost at most H_K times the minimum cost of any cut set [35].

It is also straightforward to see that the set C returned by Algorithm 2 is a cut set for P_s, which satisfies the necessary condition for LSA in Theorem 2. So the attacker can simply select $L_m = C$ as the manipulated link set. The next problem is how to inject delays to launch the attack to satisfy the conditions in (5), (6).

E. Optimize the Injected Delays to Maximize the Destroy

After determining L_m, now the decision variables become $\Delta x_j, j \in L_m$, which are the injected delays to the manipulated links. Since all the constraints in (5), (6) are linear, the injection delay optimization can be modeled as a linear programming model. We consider optimizing the delay injection to maximize the total delays of the measurement paths to cause the largest destroy to the network.

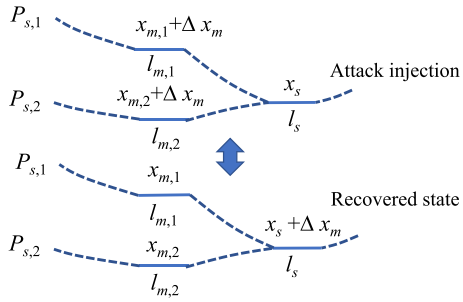

Fig. 5. An example of how the scapegoating attack is successfully launched.

Let $y'_i = \sum_{l_j \in p_i} (x_j + \Delta x_j)$ be the total delay of path p_i after delay injection, then the objective of optimization is to maximize $\sum_{i=1}^{|P|} y'_i, p_i \in P$. The linear programming model can be formulated as:

$$\max \sum_{i=1}^{|P|} y'_i, p_i \in P$$
$$\text{subject to conditions in (9)} \tag{10}$$

The linear programming problem always has an optimal solution when the parameters $\Gamma, \beta_{\max}, \beta_{\min}$ are reasonably set.

Theorem 5 (The Existence of the Optimal Attack): When $\Gamma - \sum_{l_j \in p_i} x_j \geqslant \sum_{l_s \in L_s \cap p_i} (\beta_{\max} - x_s), \forall p_i \in P_s$; and $\beta_{\min} \geqslant x_j, \forall l_j \in L_m$, the linear programming problem in (10) always have an optimal solution.

Proof: The fourth and the fifth conditions imply an implicit linear constraint $R\hat{x} = Y'$. Since $Y' = R(x + \Delta x)$, the implied condition is $R\hat{x} = R(x + \Delta x)$. Given Δx, this equation has unlimited number of solutions of $\hat{x}$ since R is not full rank. The fourth and the fifth conditions requires the solution of $\hat{x}$ satisfy $\hat{x}_i > \beta_{\max}$ for $l_i \in L_s$ and $\hat{x}_i < \beta_{\min}$ for $l_i \in L_m$.

Since all the paths in $P \setminus P_s$ should not be affected by the injected delays, $\Delta x_i = 0$ for $l_i \in P \setminus P_s$. Since L_m forms a cut set of P_s, all links in L_m are on P_s, L_m and L_s are both on P_s. Therefore, $R\hat{x} = R(x + \Delta x)$ can be reduced to $R_s \hat{x}_{[P_s]} = R_s(x + \Delta x)_{[P_s]}$ where R_s is routing matrix formed by P_s and the variables include only the delays of the links on P_s. Since L_m cuts P_s, each path in P_s contains at least one manipulated link and one scapegoat link.

Without loss of generality, we assume a scapegoat link l_s is cut by two manipulated links $l_{m,1}$ and $l_{m,2}$ as shown in Fig. 5. R_s is constructed by two probing paths in the example. After injecting delays Δx_m to manipulated links $l_{m,1}$ and $l_{m,2}$, $R_s \hat{x}_{[P_s]} = R_s(x + \Delta x)_{[P_s]}$ will hold if $\hat{x}_{m,1} = x_{m,1}, \hat{x}_{m,2} = x_{m,2}$, and $\hat{x}_s = x_s + \Delta x_m$ since:

$$R_s[\ldots, x_{m,1} + \Delta x_m, \ldots, x_{m,2} + \Delta x_m, \ldots, x_s, \ldots]^T = R_s[\ldots, x_{m,1}, \ldots, x_{m,2}, \ldots, x_s + \Delta x_m, \ldots]^T \tag{11}$$

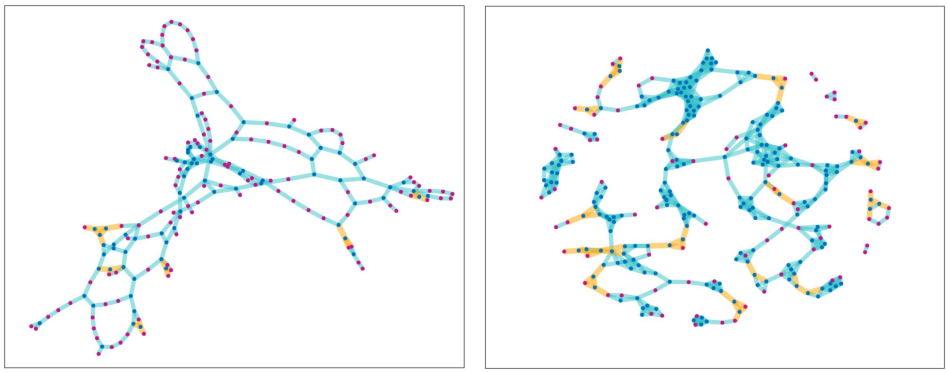

Fig. 6. The results of using the SLD Algorithm on $Cogentco$(left) and RG(right). The Links in the vulnerable link set are shown orange.

TABLE II
PARAMETERS OF TOPOLOGIES

Network	L($\mathcal{G}$)	V($\mathcal{G}$)	Monitors	Paths
AttMpls	56	25	5	40
Surfnet	68	50	32	496
TataNld	186	145	89	3916
Cogentco	243	197	120	7140
BA	395	200	74	2701
RG	677	300	82	1741
ER	1215	500	61	1830

So we only need to make $\Gamma - \max\{y_1, y_2\} \geq \Delta x_m \geq \beta_{\max} - x_s$, conditions 4,5 would be satisfied to make l_s a scapegoat. When $\Gamma - \sum_{l_j \in p_i} x_j \geqslant \sum_{l_s \in L_s \cap p_i} (\beta_{\max} - x_s), \forall p_i \in P_s$, enough delays can be injected to make $\hat{x}_i > \beta_{\max}$ for $l_i \in L_s$, while the path delay doesn't exceed Γ. Therefore, the feasible solution exists and is bounded, guaranteeing the existence of the optimal attack. □

Experiments are conducted on both real network datasets and simulated network typologies to evaluate the validity and performance of the proposed attacking methods.

VI. PERFORMANCE EVALUATION

A. Methodology

1) Network Topology: We choose real network topologies from public datasets and three widely used synthetic topologies in network tomography studies [9], [20], [21]. The parameters of these topologies are shown in Table II.

a) Four real network topologies from the Internet Topology Zoo [36] are used: AttMpls, Surfnet, TataNld, and Cogentco. They are PoP (Point of Presence)-level topologies and are manually traced from operator provided network maps.

b) Three kinds of synthetic network topologies are generated to evaluate the algorithms in large-scale networks.

- *Barabási – Albert (BA) graphs [37]:* The BA graph is frequently used to model many naturally occurring networks, such as the Internet and social networks. In order to generate a BA graph, we begin with a small connected graph $\mathcal{G} := (v_1, v_2, v_3, v_4, v_1v_2, v_1v_3, v_1v_4)$ and add nodes sequentially. A BA graph of 200 nodes is used.

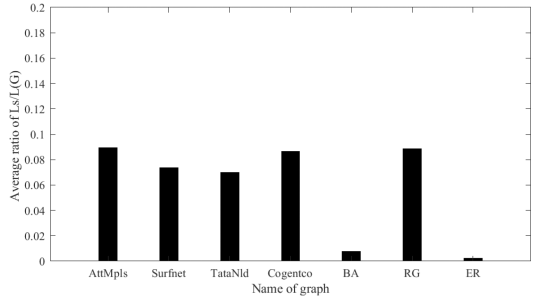

Fig. 7. The average rate of L_s in each topology.

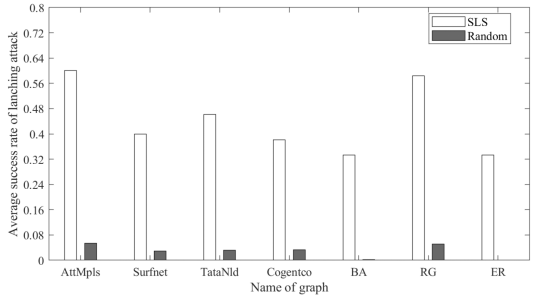

Fig. 8. The average success rate of launching scapegoating attack of two algorithms.

- *Random geometric (RG) graph [38]:* The RG graph is used to model the topology of wireless ad hoc networks. By randomly distributing nodes in a unit square and connecting each pair of nodes by a link if their distance is not greater than a threshold, we can generate a random graph. An RG graph of 300 nodes is used.
- *Erdös – Rényi (ER) graph [39]:* The ER graph denotes a random graph generated by independently connecting each pair of nodes by a link with a fixed probability p. The probability is set to be 0.014 in the simulations. An ER graph of 500 nodes is used.

For each topology, the monitor placement algorithm [20] is applied for selecting candidate monitors. After the selection of the monitors, Dijkstra's algorithm and Yen's algorithm [31] are used to find multiple shortest paths for generating the probing

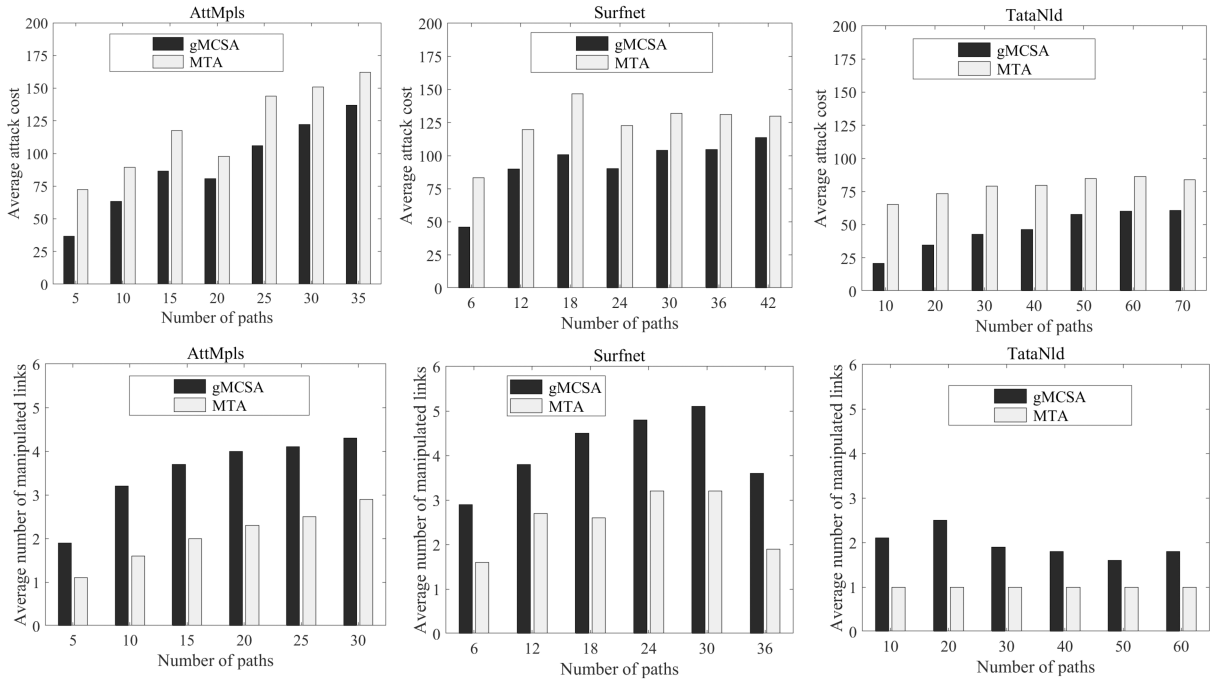

Fig. 9. The average attack cost of gMCSA and MTA (top 3); the average number of manipulated links of gMCSA and MTA (bottom 3).

paths. Then the maximum linearly independent path set P is used as the probing path set.

2) Parameter Setting: The normal delay in each link is set to 10 ms. A link is considered 'normal' if its delay is within 150 ms. To avoid being considered as a 'problematic' link, the maximum delay at a link is 2000 ms, i.e., $\beta_{\max} = 2000$. We assume an attacker can successfully inject delay if it selects the link set L_m. The attack cost of each link is a random value between 0 and 100. The cost to manipulate a link represents the difficulty of controlling a link, such as gaining backdoor access to an associated device [34].

3) Benchmarks: We compare the proposed scapegoat link detection (SLD) algorithm with random scapegoat link selection. By calculating the average success rate of launching the scapegoating attack, we derive the performance of the two methods. The performances of gMCSA are compared with that of a state-of-the-art algorithm reported in the literature, i.e., Greedy Minimum traversal attack (MTA) [21].

The following performances are compared between gMCSA and MTA:

a) the average attack cost;

b) the average number of manipulated links;

c) the average total performance degradation in the case of different number of paths.

The total performance degradation is the total amount of delays injected by the attacker over all the paths.

B. Results of Locating the Vulnerable Link Set

The scapegoat link detection (SLD) algorithm is evaluated on different types of network topologies. Fig. 6 illustrates two sample results of using the SLD Algorithm. The left(right) one shows the usage of the SLD Algorithm on AttMpls(Cogentco) to locate the vulnerable link set.

The orange links in Fig. 6 denote the links in L_s. The red vertices in Fig. 6 are the selected monitors [20]. The routing paths are not shown, for they are highly overlapped and cannot be seen clearly. By checking L_s, we find out that their link values are not identifiable from the routing matrix. The results also show that even though we use an efficient monitor placement and path construction algorithm, there is still room for launching the scapegoating attack when the network is not identifiable. Fig. 7 gives the average ratio of the size of scapegoat link set over the number of total links, i.e., $\frac{|L_s|}{|L|}$ in each topology. The SLD algorithm finds the candidates of scapegoat links and effectively improves the success rate of the scapegoating attack. Fig. 8 shows the average success rate of launching the scapegoating attack to the potential scapegoat link set. We observe that the SLD algorithm performs significantly better than the random selection algorithm. The success rates of SLD are all higher than 30% in different topologies, and the success rates of the random algorithm are all less than 8% in different topologies. It is almost impossible in BA and ER graphs to successfully launch scapegoating attacks by random scapegoat link selection. Most links in these two graphs are in triconnected components, which shows the relationship between topology structure and identifiability.

The results show that by finding the unobserved cut set, the SLD algorithm can significantly improve the success rate of launching the scapegoating attack. The failure case is that we cannot find unobserved cut set to launch the attack. Therefore, the SLD algorithm is also helpful for finding security holes in networks running network tomography. We will discuss how to remedy scapegoating attack risks in future work.

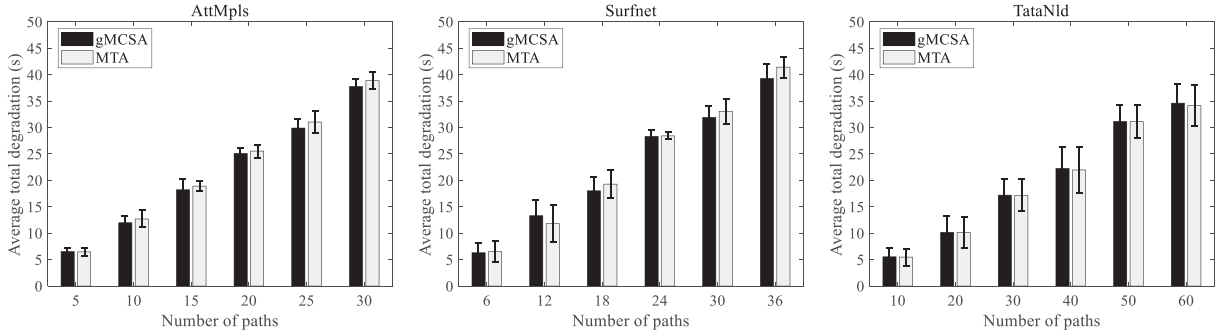

Fig. 10. The average total performance degradation of the two algorithms.

C. Performance of Scapegoating Attack

The attacking cost of gMSCA and MTA is compared by repeatedly launching the scapegoating attack thirty times using differently generated monitor deployments and routing matrices. Note that both gMSCA and MTA select the attacking links from the candidate sets that satisfy the scapegoat attacking conditions, i.e., unobserved cut set to the monitoring paths passing through the scapegoat links. MTA prefers to select the link set with the minimum traversal times.

We evaluate the average attack cost of each algorithm in the case of the different number of paths. The results are shown in Fig. 9. In this experiment, gMCSA performs much better than MTA regarding the performance of reducing the attacking cost. The average attack cost of gMCSA is significantly lower than that of MTA in all evaluated cases. The attack cost represents the difficulty of controlling links, so gMCSA is a more efficient algorithm than MTA. We also compare the average number of manipulated links of the two algorithms. In particular, we notice that gMCSA would select more manipulated links to get the lowest attack cost, while the MTA algorithm only needs to traverse the scapegoat paths. This is in line with our previous discussion.

Fig. 10 shows that both algorithms can cause significant performance degradation to the networks. gMCSA and MTA cause similar performance degradation while gMCSA achieves this using low attacking cost. But note that MTA [21] requires the manipulated links to inject different delays in different paths, which is hard to be realized. Therefore, gMCSA deserves more attention. The evaluations show that it is possible to launch LSA successfully and inject significant delays on paths without being discovered by network tomography.

VII. Conclusion

This article investigates the treats of link-based scapegoating attack (LSA) for non-identifiable networks running network tomography. We firstly provide an efficient scapegoat link detection (SLD) algorithm to locate the vulnerable link sets so that the scapegoat candidates L_s can be found. Then necessary and sufficient conditions to successfully launch LSA are proposed. It requires to find the unobserved cut set for the probing paths passing through L_s. Then a minimum cost scapegoating attack problem (MCSA) is proposed, and an approximation algorithm with H_k approximation ratio is presented. Theoretical analysis shows the existence of the optimal attack, and evaluations on both synthetic topologies and real network topologies demonstrate the effectiveness of locating L_s and launching the attacks. In future work, we will further study how to defend the scapegoating attack to improve the security of network tomography.

References

[1] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy, "Bandwidth estimation: Metrics, measurement techniques, and tools," *IEEE Netw.*, vol. 17, no. 6, pp. 27–35, Nov./Dec. 2003.
[2] Graphical Traceroute Software | PingPlotter, 2021. [Online]. Available: https://www.pingplotter.com/
[3] Cisco IOS NetFlow, 2021. [Online]. Available: https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html
[4] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Pub. Amer. Stat. Assoc.*, vol. 91, no. 433, pp. 365–377, 1996.
[5] T. Bu, N. Duffield, F. L. Presti, and D. Towsley, "Network tomography on general topologies," in *ACM Sigmetrics Int. Conf. Meas. Model. Comput. Syst.*, 2002, pp. 21–30.
[6] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Monitor placement for maximal identifiability in network tomography," in *Proc. IEEE Conf. Comput. Commun.*, 2014, pp. 1447–1455.
[7] N. Duffield, F. Lo Presti, V. Paxson, and D. Towsley, "Network loss tomography using striped unicast probes," *IEEE/ACM Trans. Netw.*, 14, no. 4, pp. 697–710, Aug. 2006.
[8] A. Chen, J. Cao, and T. Bu, "Network tomography: Identifiability and fourier domain estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 12, pp. 6029–6039, 2010.
[9] S. Tati, S. Silvestri, T. He, and T. L. Porta, "Robust Network Tomography in the Presence of Failures," in *Proc. IEEE 34th Int. Conf. Distrib. Comput. Syst.*, 2014, pp. 481–492.
[10] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Identifiability of link metrics based on end-to-end path measurements," in *Proc. Conf. Internet Meas. Conf.*, 2013, pp. 391–404.
[11] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, "Efficient identification of additive link metrics via network tomography," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst.*, 2013, pp. 581–590.
[12] Y. Qiao, J. Jiao, Y. Rao, and H. Ma, "Adaptive path selection for link loss inference in network tomography applications," *PLos One*, vol. 11, no. 10, Art. no. e0163706, Oct. 2016.
[13] Y. Gu, G. Jiang, V. Singh, and Y. Zhang, "Optimal probing for unicast network delay tomography," in *Proc. IEEE Int. Conf. Comput. Commun.*, 2010, pp. 1–9.
[14] Y. Xia and D. Tse, "Inference of link delay in communication networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 12, pp. 2235–2248, Dec. 2006.
[15] A. Gopalan and S. Ramasubramanian, "On identifying additive link metrics using linearly independent cycles and paths," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 906–916, Jun. 2012.
[16] H. X. Nguyen and P. Thiran, "The boolean solution to the congested IP link location problem: Theory and practice," in *Proc. IEEE 26th Int. Conf. Comput. Commun.*, 2007, pp. 2117–2125.

[17] N. Alon, Y. Emek, M. Feldman, and M. Tennenholtz, "Economical graph discovery," *Operations Res.*, 62, no. 6, pp. 1236–1246, Oct. 2014.
[18] T. He, L. Ma, A. Gkelias, K. K. Leung, A. Swami, and D. Towsley, "Robust monitor placement for network tomography in dynamic networks," in *Proc. IEEE 35th Annu. Int. Conf. Comput. Commun.*, 2016, pp. 1–9.
[19] L. Ma, T. He, A. Swami, D. Towsley, and K. K. Leung, "On optimal monitor placement for localizing node failures via network tomography," *Performance Eval.*, vol. 91, pp. 16–37, Sep. 2015.
[20] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Inferring link metrics from End-to-end path measurements: Identifiability and monitor placement," *IEEE/ACM Trans. Netw.*, 22, no. 4, pp. 1351–1368, Aug. 2014.
[21] C. Chiu and T. He, "Stealthy DGoS attack: DeGrading of service under the watch of network tomography," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 367–376.
[22] S. Zhao, Z. Lu, and C. Wang, "When seeing isn't believing: On feasibility and detectability of scapegoating in network tomography," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst.*, 2017, pp. 172–182.
[23] S. Zhao, Z. Lu, and C. Wang, "Measurement integrity attacks against network tomography: Feasibility and defense," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 6, pp. 2617–2630, Nov./Dec. 2021.
[24] J. Sen, S. Koilakonda, and A. Ukil, "A mechanism for detection of cooperative black hole attack in mobile Ad Hoc networks," in *Proc. IEEE 2nd Int. Conf. Intell. Syst., Modelling Simul.*, 2011, pp. 338–343.
[25] B. Sharma, "A distributed cooperative approach to detect gray hole attack in MANETs," in *Proc. 3rd Int. Symp. Women Comput. Inform.*, 2015, pp. 560–563.
[26] J. Zhao, R. Govindan, and D. Estrin, "Sensor network tomography: Monitoring wireless sensor networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 1, p. 64, 2002.
[27] M. F. Shih and A. Hero, "Unicast inference of network link delay distributions from edge measurements," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process.*, 2001, pp. 3421–3424.
[28] A. Adams et al., "The use of End-to-end multicast measurements for characterizing internal network behavior," *IEEE Commun. Mag.*, vol. 38, no. 5, pp. 152–159, May 2000.
[29] B. Xi, G. Michailidis, and V. N. Nair, "Estimating network loss rates using active tomography," *J. Amer. Statist. Assoc*, vol. 101, pp. 1430–1438, 2006.
[30] W. Dong, Y. Gao, W. Wu, J. Bu, C. Chen, and X. Li, "Optimal monitor assignment for preferential link tomography in communication networks," *IEEE/ACM Trans. Netw.*, 25, no. 1, pp. 210–223, Feb. 2017.
[31] T. Pepe and M. Puleri, "Network Tomography: A novel algorithm for probing path selection," *IEEE Int. Conf. Commun.*, 2015, pp. 5337–5341.
[32] C. Feng, L. Wang, K. Wu, and J. Wang, "Bound-based network tomography with additive metrics," in *Proc. IEEE Int. Conf. Comput. Commun. Conf. Comput. Commun.*, 2019, pp. 316–324.
[33] J. Hopcroft and R. Tarjan, "Dividing a graph into triconnected components," *SIAM J. Comput.*, vol. 2, pp. 135–158, Sep. 1973.
[34] L. Constantin, "Attackers slip rogue, backdoored firmware onto Cisco routers | PCWorld," 2015. [Online]. Available: https://www.pcworld.com/article/423575/attackers-install-highly-persistent-malware-implants-on-cisco-routers.html
[35] V. V. Vazirani, *Approximation Algorithms*. Berlin, Heidelberg: Springer, 2001.
[36] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.
[37] R. Albert and A. L. Barabasi, "Statistical mechanics of complex networks," *Rev. Modern Phys.*, vol. 74, pp. 47–97, Jan. 2002.
[38] M. Penrose, *Random geometric graphs*. London, U.K.: Oxford Univ. Press, 2003.
[39] P. Erdós and A. Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, pp. 17–60, Jan. 1960.

Xiaojia Xu received the B.S degrees from the Department of Computer Science, Renmin University of China, Beijing, China, in 2020. She is currently working toward the Ph.D. degree with the School of Information, Renmin University of China. Her research interests include algorithm design and analysis, wireless networks and operations research.

Yongcai Wang (Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Automation Sciences and Engineering, Tsinghua University, Beijing, China, in 2001 and 2006 respectively. From 2007 to 2009, he was Associated Researcher with NEC Labs, China. From 2009 to 2015, he was a Research Scientist with the Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University. He was a Visiting Scholar with Cornell University, Ithaca, NY, USA, in 2015. He is currently Associate Professor with the Department of Computer Sciences, Renmin University of China, Beijing. His research interests include intelligent network, combinatorial optimization algorithms and applications.

Lanling Xu is currently working toward the master's degree with the School of Information, Renmin University of China, Beijing, China, majoring in computer science and technology. Her research interests include deep learning, recommender system and natural language processing.

Deying Li received the M.S. degree in mathematics from the Huazhong Normal University, Wuhan, China. in 1988, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2004. She is currently a Professor with the Department of Computer Science, Renmin University of China, Beijing, China. Her research includes wireless networks, mobile computing and algorithm design and analysis.