Stealthy DGoS Attack: DeGrading of Service under the Watch of Network Tomography

Cho-Chun Chiu, *Student Member, IEEE* and Ting He, *Senior Member, IEEE*

***Abstract*—Network tomography is a powerful tool to monitor the internal state of a closed network that cannot be measured directly, with broad applications in the Internet, overlay networks, and all-optical networks. However, existing network tomography solutions all assume that the measurements are trust-worthy, leaving open how effective they are in an adversarial environment with possibly manipulated measurements. To understand the fundamental limit of network tomography in such a setting, we formulate and analyze a novel type of attack that aims at maximally degrading the performance of targeted paths without being localized by network tomography. By analyzing properties of the optimal attack strategy, we formulate novel combinatorial optimizations to design the optimal attack strategy, which are then linked to well-known NP-hard problems and approximation algorithms. As a byproduct, our algorithms also identify approximations of the most vulnerable set of links that once manipulated, can inflict the maximum performance degradation. Our evaluations on real topologies demonstrate the large potential damage of such attacks, signaling the need of new defenses.**

***Index Terms*—Network tomography, Denial of Service attack, combinatorial optimization, approximation algorithm.**

I. Introduction

Timely and accurate knowledge of network internal state (e.g., link delays/jitters/loss rates/bandwidths) is essential for many network management functions such as traffic engineering, load balancing, and service placement, which actively adapt control parameters such as the routes, the rates, and even the destinations (e.g., via service placement) according to the current network state.

Traditionally, network administrators obtain the network state by directly measuring internal network elements through local support (e.g., SNMP agents) or special diagnostic tools (e.g., traceroute). This approach has the limitation that it requires the support of internal network devices, e.g., to run SNMP agent or respond to ICMP probes, which has severe limitations in networks where such support is unreliable [2], [3], [4] or unavailable [5], [6].

Network tomography [7] provides a powerful approach for monitoring the internal state of closed networks. Instead of directly measuring the internal elements, network tomography infers the states of these elements (e.g., link delays) from end-to-end measurements (e.g., path delays) between special nodes participating in monitoring, referred to as *monitors*. As network tomography only requires the cooperation from monitors, it has broad applications in monitoring networks where only a subset of nodes cooperate, e.g. the Internet [2], [3], [4], overlay networks [8], and all-optical networks [5], [6].

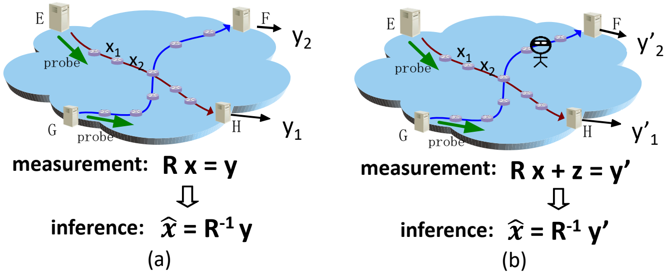


Fig. 1. (a) network tomography in benign setting; (b) network tomography in adversarial setting.

Despite substantial research on network tomography, most existing solutions hinge on a fundamental assumption: *the measurements correctly reflect the performance of measurement paths*. Consider the canonical application of inferring additive link metrics (e.g., delays, jitters, log-success rates) from the sum metrics on measurement paths. As illustrated in Fig. 1 (a), normally the measured path metrics will equal the sum of link metrics on each path, yielding a linear observation model: $R\mathbf{x} = \mathbf{y}$, where $\mathbf{x} = (x_j)_{l_j \in L}$ is the column vector of unknown link metrics (L: set of links), $\mathbf{y} = (y_i)_{p_i \in P}$ is the column vector of measured path metrics (P: set of measurement paths), and $R = (r_{ij})_{p_i \in P, l_j \in L}$ is the *measurement matrix* with $r_{ij} \in \{0, 1\}$ indicating whether path p_i traverses link l_j. Network tomography infers the link metrics by "inverting" the observation model, i.e., solving for $\widehat{\mathbf{x}}$ that satisfies $R\widehat{\mathbf{x}} = \mathbf{y}$ (the solution may not be unique).

However, if some links are controlled by an attacker (referred to as *compromised links*) as illustrated in Fig. 1 (b), then the attacker can manipulate the measurements on paths traversing these links, e.g., by introducing additional delays, jitters, or losses. This yields a modified observation model: $R\mathbf{x}+\mathbf{z} = \mathbf{y}'$, where $\mathbf{y}'$ is the vector of observed path metrics under the attack, and $\mathbf{z} = (z_i)_{p_i \in P}$ is the vector of *manipulations* controlled by the attacker. For example, the attacker can be a malicious Internet Service Provider (ISP) [9] that tries to attack a targeted content provider, whose paths to clients are modeled by P, from a set of links it controls in the public Internet. Another example is a hacker that launches an attack on a targeted institutional network by remotely controlling its backdoor-infected routers [10]. Note that the modified observation model is different from $R(\mathbf{x}+\mathbf{z}) = \mathbf{y}'$, as the attacker can manipulate packets on different paths differently at the same link, e.g., delaying packets belonging to one path but not delaying packets belonging to another path. An unsuspecting network tomography algorithm will try to explain the measurements according to the original observation model by trying to find $\widehat{\mathbf{x}}$ satisfying $R\widehat{\mathbf{x}} = \mathbf{y}'$. This can cause many issues, such as lack of feasible solutions [11] and incorrect fault diagnosis [12].

C. Chiu (cuc496@psu.edu) and T. He (tzh58@psu.edu) are with the Pennsylvania State University.

A preliminary version of this work was presented at INFOCOM'20 [1].

The research was sponsored by NSF under Award Number 1813219.

In this work, we aim to understand the fundamental limit of a stealthy attacker in *maximally degrading the performance of end-to-end communications without being localized by network tomography*. Such understanding will not only quantify the limitation of existing network tomography algorithms but also provide insights for the design of defense mechanisms.

A. Related Work

Since introduced by Vardi [13], network tomography has expanded to a rich family of network monitoring techniques that infer network internal characteristics from external measurements [7], [14]. Early works focused on *best-effort solutions*, which tried to find the most likely network state from given measurements, obtained by unicast [15], [16], [17], [18], multicast [19], [20], [21], [22], [23], [24], and their variations (e.g., bicast [25], flexicast [26], and back-to-back unicast [27], [28], [24]). After observing that an arbitrary set of measurements is frequently insufficient for identifying all the link metrics [29], [16], [30], [8], [31], later works aimed at either reducing ambiguity by imposing a tie breaker (e.g., [17], [18], [32]) or relaxing the objective (e.g., [8], [33], [34]), or ensuring identifiability by carefully designing the monitor locations and the paths to measure [35], [36], [37], [38], [39], [6], [40], [41], [42], [43]. All these works assume a *benign setting*, where the links behave consistently.

In contrast, very few works have considered network tomography in an *adversarial setting*, where links can behave inconsistently for different paths. In [11], the problem is tackled in the context of a non-neutral network, where some links can discriminate packets sent on different paths. In [12], the problem is tackled in the context of an attacker that can manipulate the measurements traversing malicious nodes, with a primary goal of scapegoating certain benign links as the cause of poor performance. While our problem setting is similar to [12], our results differ significantly as explained in Section II-C.

B. Summary of Contributions

The main contributions of this work are:

1) We formulate a novel attack, called *stealthy DeGrading of Service (DGoS) attack*, that aims at maximally degrading the performances of end-to-end communications by manipulating the performances of compromised links, without letting these links localized by network tomography.

2) To understand the fundamental limit of this attack, we develop algorithms to explicitly design which links to compromise and how to manipulate the performances of these links. We show that selecting which links to compromise is a novel combinatorial optimization problem that is NP-hard. By linking this problem to known NP-hard problems, we leverage existing algorithms to achieve guaranteed approximation.

3) We further consider a budget constraint on the cost of compromising links. We show that the constrained link selection problem is another novel combinatorial optimization problem that is also NP-hard. By relaxing the objective function, we again link this problem to a known NP-hard problem that allows us to leverage an existing approximation algorithm.

4) Our evaluations on real topologies show that the proposed attack can significantly degrade the communication performance (by injecting 4–30 seconds of delay per path) without exposing the compromised links to network tomography.

Roadmap. Section II formulates our problem. Section III designs the attack in the unconstrained case, which is evaluated in Section IV. Section V addresses the constrained case. Finally, Section VI concludes the paper.

II. Problem Formulation

A. Network Model

We model the network monitored by network tomography as an undirected graph $\mathcal{G} = (N, L)$, where N is the set of nodes and L the set of links. Each link $l_j \in L$ is associated with an unknown metric x_j that describes its performance (e.g., average link delay). We assume that these link metrics are *additive*, i.e., the metric of a path equals the sum of its link metrics, which is a canonical model representing important performance metrics including delays, jitters, log-success rates, and many other statistics.

B. Network Tomography Model

Suppose that a set of users of the above network (or their proxy) send traffic through $\mathcal{G}$ along a set of paths P, and use network tomography to monitor the received performances at individual links. Using network tomography to monitor the performance of individual links from path-level measurements is a well-established technique that is particularly relevant to the Internet [14], due to the opaque nature of the ISP networks to the providers of host-based distributed systems and applications. In host-based distributed systems such as virtual private networks (VPNs) and content distribution networks (CDNs), as well as adaptive applications such as streaming media and multiplayer gaming, a tomography-based overlay monitoring system can detect periods of degraded performance within seconds, thus facilitating informed adaptation of overlay paths and communication patterns [44].

Let $R = (r_{ij})_{p_i \in P, l_j \in L}$ be the matrix representation of P, called the *measurement matrix*, where $r_{ij} \in \{0, 1\}$ indicates if path p_i traverses link l_j. Let $\mathbf{r}_i = (r_{ij})_{l_j \in L}$ be the i-th row in R. Given the measured path metrics $\mathbf{y} = (y_i)_{p_i \in P}$, network tomography seeks to find a solution $\widehat{\mathbf{x}}$ to the link metrics that can explain the measurements, i.e., $R\widehat{\mathbf{x}} = \mathbf{y}$.

We note that the solution is generally non-unique as R may not be full-column-rank. This issue, known as the *lack of identifiability*, has been widely recognized [29], [16], [30], [8], [31]. Instead of making a limiting assumption that R must be full-column-rank as in [12], we allow an arbitrary R, and consider a generic network tomography solver that can compute the set of all feasible solutions.

C. Attack Model

1) Threat Model: Suppose that an attacker attempts to degrade the performance of P by manipulating the performances of compromised links. Let $L_m \subseteq L$ denote the set of *compromised links* and $L_n = L \backslash L_m$ the set of *uncompromised links*. Accordingly, the paths $P_m \subseteq P$ traversing at least one compromised link are called *compromised paths*, and the remaining paths $P_n = P \setminus P_m$ are called *uncompromised paths*. The attacker can only control the compromised links.

One possible attack scenario is an ISP-based attacker that tries to degrade the Quality of Service (QoS) of a targeted content provider as studied in the context of network neutrality [11], except that the ISP itself is not involved in the attack (hence only the links compromised by the attacker will participate). In this case, P contains all the paths within this ISP network that are between the gateway router to the content provider and the other gateway routers. In this scenario, we model a more intelligent adversary than [11] that avoids causing infeasibility of the network tomography problem and thus evades detection by the existing detector in [11]. Similar scenario exists when targeting a client network.

Another possible attack scenario is an attacker in a legacy underlay network that tries to degrade the performance of an overlay network used to implement state-of-the-art control algorithms [45]. In this case, P contains all the paths within the underlay network that connect the overlay nodes. In this scenario, we model a novel type of adversarial intervention that controls the forwarding performance, complementing the existing model in [45] that controls the forwarding direction.

Our model implicitly assumes that all the measurement paths monitored by network tomography are fixed and known to the attacker. Assuming fixed measurement paths is a standard assumption in network tomography, which underlies nearly all tomography-based inference algorithms. Meanwhile, while the exact set $\tilde{P}$ of paths evaluated by network tomography will not be observable to the attacker, the attacker can construct a possibly larger set P of potential measurement paths with basic knowledge of the attacked network $\mathcal{G}$, e.g., topology, routing, and ingress/egress points (e.g., gateway routers). If $\tilde{P} \subset P$, it is easy to see that a stealthy attack designed for P remains stealthy for $\tilde{P}$ (in the sense modeled by (2)). However, the effectiveness of the attack can be suboptimal due to unnecessary constraints induced by paths in $P \setminus \tilde{P}$. We will evaluate this case later (see Fig. 10).

2) Attack Optimization: Let $\mathbf{z} = (z_i)_{p_i \in P}$ denote the vector of *manipulations*, where z_i is the increment in the metric of path $p_i \in P$ caused by the attacker. It is easy to see that $\mathbf{z}$ must satisfy the following constraints [12]:

1) Only the metrics of compromised paths can be manipulated, i.e., $z_i = 0$ for any $p_i \in P_n$.
2) Path performances can only be degraded (not improved) due to manipulation, i.e., $z_i \geq 0$ for any $p_i \in P_m$.

Moreover, to stay stealthy, the attacker must preserve feasibility of the network tomography problem to hide the presence of artificial manipulations, i.e., after the manipulations, there must exist at least one solution $\widehat{\mathbf{x}}$ that satisfies $R\widehat{\mathbf{x}} = R\mathbf{x} + \mathbf{z}$. In addition, he must protect the compromised links from detection. As a concrete example, we consider *threshold-based bad link detection*, where the state δ_j (1: bad, 0: good) for link l_j is inferred as

$$\delta_j = \begin{cases} 1 & \text{if } \widehat{x}_j > \tau, \\ 0 & \text{o.w.,} \end{cases} \tag{1}$$

Here, τ denotes the detection threshold (e.g., the maximum normal link delay). Threshold-based detection is widely used in network monitoring systems (e.g., NetFlow Analyzer [46], OpManager [47]), and threshold-based bad link detection is a natural application for network tomography. To evade such detection, our attack model requires that among all the feasible solutions to $\widehat{\mathbf{x}}$, there must be at least one solution that does not flag any of the compromised links as bad links. In practice, there may also be an upper bound on link metrics, denoted by $\tau_{\max}$, e.g., the maximum duration a packet can be buffered at a network interface without being dropped. To avoid trivial cases, we assume that $x_j \leq \tau \leq \tau_{\max}$ for all $l_j \in L$.

We formulate the attacker's goal as the following optimization, called the *stealthy DeGrading of Service (DGoS) attack*:

$$\max_{L_m, \widehat{\mathbf{x}}} \sum_{p_i \in P_m} \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) \tag{2a}$$

$$\text{s.t. } \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) = 0, \quad \forall p_i \in P_n, \tag{2b}$$

$$\mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) \geq 0, \quad \forall p_i \in P_m, \tag{2c}$$

$$\tau_{\max} \geq \widehat{x}_j \geq 0, \quad \forall l_j \in L_n, \tag{2d}$$

$$\tau \geq \widehat{x}_j \geq 0, \quad \forall l_j \in L_m, \tag{2e}$$

$$L_m \subseteq L. \tag{2f}$$

This is an optimization of L_m and $\widehat{\mathbf{x}}$, where L_m specifies the links to compromise, and $\widehat{\mathbf{x}}$, denoting (one of the feasible solutions to) the inferred link metrics, is used to compute the *actual manipulations* $\mathbf{z}$ to inject onto the paths by

$$\mathbf{z} = R(\widehat{\mathbf{x}} - \mathbf{x}). \tag{3}$$

Computing the manipulations by (3) automatically ensures feasibility of the network tomography problem. Note that this does not require the compromised links to behave consistently across paths, as illustrated in Fig. 2.

In words, the objective (2a) is to maximize the total performance degradation on paths in P, measured by the increase in the sum path metric. Constraints (2b,2c) ensure that manipulations are feasible, i.e., only performed on compromised paths to degrade the performance. Constraint (2e) ensures that the attack cannot be localized by network tomography, as all the compromised links perform normally according to the inferred link metrics. Note that $\widehat{\mathbf{x}}$ only represents the link performances perceived by network tomography, which are generally not the same as the actual link performances. As is shown later, an intelligent attacker can leverage this difference to inject performance degradation on end-to-end communications at compromised links, while causing network tomography to blame the degradation on some uncompromised links (thus keeping the compromised links undetected). Note that this attack is only designed to evade threshold-based detection; for more sophisticated detection systems (e.g., those examining the distribution of link metrics), randomization of the upper bounds in (2d)–(2e) will be needed to generate a plausible $\widehat{\mathbf{x}}$.

Remark 1: The above formulation is based on an optimistic constraint, i.e., there exists a possible solution to the link metrics that does not flag any of the compromised links as bad links, which ensures that the provider cannot say for sure that any of the attacker-controlled links is the cause of poor end-to-end performance. In the case of rank-deficient R, there will be other solutions that possibly flag some of these links. One way of achieving stronger stealthiness is to include additional constraints to ensure that the desired $\widehat{\mathbf{x}}$ will be the solution selected by network tomography (e.g., flagging the fewest links among all possible solutions), which requires additional knowledge of the adopted network tomography algorithm. The strongest stealthiness is achieved by requiring that no feasible solution to $R\widehat{\mathbf{x}} = R\mathbf{x} + \mathbf{z}$ will flag any of the compromised links. As a first step towards understanding the potential damage of DGoS attacks, we will focus on the formulation in (2) and empirically evaluate its stealthiness under a practical tomography-based detector (see Fig. 11), while leaving the detailed study of other formulations to future work. Note that in the case of full-column-rank R as assumed previously [12], these formulations become the same.

Remark 2: For clarity, we will present all the results from the perspective of an attacker. However, our results can also be interpreted from the perspective of a network provider that uses network tomography to validate link performances. In this case, the optimal objective value of (2) reveals the maximum damage that an in-network adversary can inflict on end-to-end communications without being localized, and the corresponding decision variables (particularly L_m) specify the most vulnerable links that can be manipulated to inflict the maximum damage. Thus, our results can be used to analyze network vulnerability and recommend high-value links to protect.

Remark 3: Our work differs fundamentally from the existing works [11], [12] that also considered network tomography in an adversarial setting. Specifically, although [11] proposed an algorithm to detect links that behave inconsistently on different paths (i.e., non-neutral links), the algorithm only works when the inconsistent links cause infeasibility of the network tomography problem, and thus cannot handle our attack model that always ensures feasibility. Moreover, although the attack model studied in [12] is conceptually similar to ours in that the attacker also tries to fool network tomography while degrading path performances, their results substantially differ from ours in that: (i) the attacker in [12] is required to mislead network tomography to detect certain uncompromised links as bad links, while we do not impose such constraints; (ii) [12] assumes the measurement matrix to be full-column-rank, which is frequently violated in practice [29], [16], [30], [8], [31], while we do not make such an assumption; (iii) most importantly, [12] assumes that the set L_m of compromised links is given, while we treat it as a decision variable, which allows us to model a more intelligent attacker that strategically places its attack. In fact, as is shown later, the selection of L_m significantly impacts the capability of an attack and is thus the focus of our work. Our solutions on optimizing L_m can also be used to identify the most vulnerable links to protect from a network provider's perspective.

D. Example

Consider the example in Fig. 2 (a). Suppose that before the attack, each link has a delay of 10 ms, $\tau = 150$ ms, and $\tau_{\max} = 2000$ ms. Fig. 2 (b) shows the optimal manipulations under an intuitive selection of L_m—compromising all the links. In this case, the attack can cause 2240 ms of extra delay in total, by injecting a delay of z_i onto path p_i at some of the compromised links traversed by p_i. Fig. 2 (c) shows the optimal manipulations under another selection of L_m, which, although having fewer compromised links, is able to cause 15190 ms of extra delay, as the uncompromised links l_3 and l_5 can be used to explain the large delays of paths p_1, p_2, p_4, p_5, p_6 to network tomography without exposing the compromised links. Note that the inferred link metrics can differ from the actual metrics, and the compromised links can behave inconsistently across paths, e.g., in Fig. 2 (c), link l_1 injects no more than 280 ms of delay onto p_3 but 4120 ms of delay onto p_2. This example shows that DGoS attack can cause large damage without being localized, and the amount of damage critically depends on the selection of L_m.

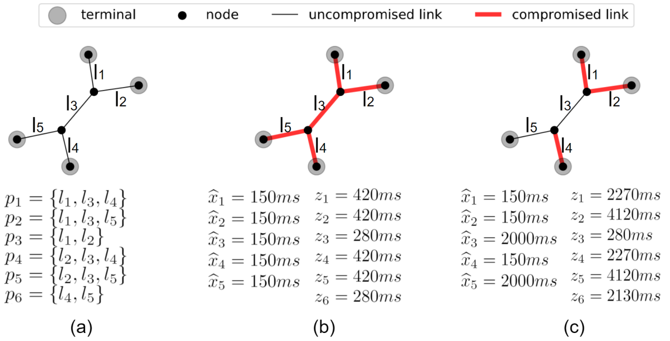

● terminal
● node
— uncompromised link
▬ compromised link
l1
l2
l3
l4
l5
$p_1 = \{l_1, l_3, l_4\}$
$p_2 = \{l_1, l_3, l_5\}$
$p_3 = \{l_1, l_2\}$
$p_4 = \{l_2, l_3, l_4\}$
$p_5 = \{l_2, l_3, l_5\}$
$p_6 = \{l_4, l_5\}$
(a)
$\widehat{x}_1 = 150ms$ $z_1 = 420ms$
$\widehat{x}_2 = 150ms$ $z_2 = 420ms$
$\widehat{x}_3 = 150ms$ $z_3 = 280ms$
$\widehat{x}_4 = 150ms$ $z_4 = 420ms$
$\widehat{x}_5 = 150ms$ $z_5 = 420ms$
$z_6 = 280ms$
(b)
$\widehat{x}_1 = 150ms$ $z_1 = 2270ms$
$\widehat{x}_2 = 150ms$ $z_2 = 4120ms$
$\widehat{x}_3 = 2000ms$ $z_3 = 280ms$
$\widehat{x}_4 = 150ms$ $z_4 = 2270ms$
$\widehat{x}_5 = 2000ms$ $z_5 = 4120ms$
$z_6 = 2130ms$
(c)

Fig. 2. Example: (a) input, (b) optimal manipulations under one L_m, (c) optimal manipulations under another L_m.

III. Optimal Attack Strategy

Although (2) is a joint optimization of both L_m and $\widehat{\mathbf{x}}$, we will show that the main challenge is in optimizing L_m, which can be reduced to a novel variation of the minimum cut problem.

A. Optimizing $\widehat{\mathbf{x}}$ under Given L_m

Given the set of compromised links L_m, (2) is a *linear program (LP)* in $\widehat{\mathbf{x}}$ that can be solved in polynomial time by standard LP solvers, and the result gives the optimal manipulation vector (under the given L_m) by (3). Nevertheless, there are several simplifications that can be used to speed up the solution for large networks.

First, we observe that constraint (2c) has no effect on the optimal solution, as it only imposes a lower bound on $\widehat{\mathbf{x}}$, while the objective (2a) tries to increase $\widehat{\mathbf{x}}$. We can thus drop this constraint without changing the optimal solution to $\widehat{\mathbf{x}}$.

Furthermore, we observe that the dimension of the solution space can be reduced. To this end, we rewrite (2) after dropping constraint (2c) in a vector form:

$$\max_{\widehat{\mathbf{x}}} \mathbf{1}_{|P_m|} R_m(\widehat{\mathbf{x}} - \mathbf{x}) \tag{4a}$$

$$\text{s.t. } R_n\widehat{\mathbf{x}} = R_n\mathbf{x}, \tag{4b}$$

$$\boldsymbol{\phi} \geq \widehat{\mathbf{x}} \geq \mathbf{0}, \tag{4c}$$

where $\mathbf{1}_{|P_m|}$ is the $1 \times |P_m|$ vector of 1's, $R_m = (\mathbf{r}_i)_{p_i \in P_m}$ and $R_n = (\mathbf{r}_i)_{p_i \in P_n}$ are the sub-measurement matrices representing all the compromised/uncompromised paths, respectively, and $\boldsymbol{\phi} := (\phi_j)_{l_j \in L}$ is the vector of upper bounds on $\widehat{x}_j$ in (2d,2e), i.e.,

$$\phi_j := \begin{cases} \tau & \text{if } l_j \in L_m, \\ \tau_{\max} & \text{if } l_j \in L_n. \end{cases} \tag{5}$$

The "$\geq$" in (4c) means element-wise $\geq$.

To reduce the dimension for optimization (4), we perform a change of variable as follows. Since $x_j \le \tau$ ($\forall l_j \in L$), it is easy to see that $\widehat{\mathbf{x}} = \mathbf{x}$ is a feasible solution to (4). Let B be a matrix whose columns form a basis of null(R_n), the null space of R_n. Let nullity(R_n) denote the *nullity* of R_n, i.e., the dimension of null(R_n). Then $\widehat{\mathbf{x}} = B\mathbf{c} + \mathbf{x}$ will always satisfy $R_n\widehat{\mathbf{x}} = R_n\mathbf{x}$ for any (nullity$(R_n) \times 1$)-vector $\mathbf{c}$. Substituting $\widehat{\mathbf{x}}$ by $B\mathbf{c} + \mathbf{x}$, (4) is transformed into:

$$\max_{\mathbf{c}} \mathbf{1}_{|P_m|} R_m B\mathbf{c} \tag{6a}$$

$$\text{s.t. } \boldsymbol{\phi} - \mathbf{x} \ge B\mathbf{c} \ge -\mathbf{x}. \tag{6b}$$

Compared to (4), the number of decision variables in (6) is reduced from the number of links to the nullity of R_n. By the *rank-nullity theorem*, rank(R_n) + nullity$(R_n) = |L|$, and hence the reduction will be significant when rank(R_n) is large, i.e., the number of linearly independent uncompromised paths is large.

B. Property of the Optimal L_m

To facilitate the optimization of L_m, we first investigate the property of the optimal solution. As is shown in Section II-D, simply compromising all the links is generally suboptimal, as the attacker will have to make all the link metrics appear normal (i.e., $\widehat{x}_j \le \tau$ for all $l_j \in L$), which limits the amount of performance degradation he can inject on each path.

Generally, compromising a link l_j can have two contradicting effects:

1) previously uncompromised paths that traverse l_j can now be controlled by the attacker, which removes some constraints of the type (2b) and hence may increase the objective value;
2) instead of constraint (2d), l_j will be subject to a tighter constraint (2e), which may decrease the objective value.

Due to these contradicting effects, it is not obvious what is the optimal set of links to compromise.

Our main result is a closed-form characterization of the optimal set of compromised links. To present this result, we introduce the following definitions.

Definition 1. Given a set of paths P, we define:

1) the *traversal number* of link l, denoted by w_l, as the number of paths in P that traverse link l;
2) a *cut* C of P as a subset of links such that every $p \in P$ traverses at least one link in C;
3) the *minimum-traversal cut* C^* of P as the cut of P with the minimum total traversal number, i.e., $\sum_{l \in C^*} w_l \le \sum_{l \in C} w_l$ for any cut C.

Theorem III.1. *The optimal set of compromised links L_m^* (i.e., the optimal solution to L_m in (2)) is the minimum-traversal cut of P.*

We will prove this theorem in two steps. **Step 1** is to show that L_m^* must be a cut of P, as otherwise the attacker will be able to improve his objective value by compromising one more link.

Lemma III.2. *Suppose that for the initial set of compromised links $L_m^{(0)}$, there is at least one uncompromised path p_{i^*}. Then there must exist an uncompromised link $l_{j^*} \in p_{i^*}$, such that compromising l_{j^*} increases the total performance degradation, i.e., $\Gamma(L_m^{(0)} \cup \{l_{j^*}\}) \ge \Gamma(L_m^{(0)})$, where $\Gamma(L')$ is the optimal objective value of (2) when $L_m = L'$.*

Proof. Let $L_m^{(0)}$ ($L_n^{(0)}$) be the initial set of compromised (uncompromised) links, $P_m^{(0)}$ ($P_n^{(0)}$) be the initial set of compromised (uncompromised) paths, and $\widehat{\mathbf{x}}^{(0)}$ be the optimal solution to $\widehat{\mathbf{x}}$ when $L_m = L_m^{(0)}$. By assumption, $p_{i^*} \in P_n^{(0)}$.

First, we observe that there must exist a link $l_{j^*} \in p_{i^*}$ for which $\widehat{x}_{j^*}^{(0)} \le \tau$, as otherwise (i.e., $\widehat{x}_j^{(0)} > \tau$ for all $l_j \in p_{i^*}$), we will have $\mathbf{r}_{i^*}\widehat{\mathbf{x}}^{(0)} > |p_{i^*}|\tau \ge \mathbf{r}_{i^*}\mathbf{x}$, where $|p_{i^*}|$ is the hop count on p_{i^*}. This contradicts with $\mathbf{r}_{i^*}\widehat{\mathbf{x}}^{(0)} = \mathbf{r}_{i^*}\mathbf{x}$ according to constraint (2b).

Next, for the above link l_{j^*}, adding a constraint $\widehat{x}_{j^*} \le \tau$ to (2) will not change the optimal solution when $L_m = L_m^{(0)}$. That is, $\widehat{\mathbf{x}}^{(0)}$ remains an optimal solution to the following optimization in $\widehat{\mathbf{x}}$

$$\max_{\widehat{\mathbf{x}}} \sum_{p_i \in P_m^{(0)}} \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) \tag{7a}$$

$$\text{s.t. } \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) = 0, \qquad \forall p_i \in P_n^{(0)}, \tag{7b}$$

$$\tau_{\max} \ge \widehat{x}_j \ge 0, \qquad \forall l_j \in L_n^{(0)} \setminus \{l_{j^*}\}, \tag{7c}$$

$$\tau \ge \widehat{x}_j \ge 0, \qquad \forall l_j \in L_m^{(0)} \cup \{l_{j^*}\}. \tag{7d}$$

Note that we can omit constraint (2c) as explained in Section III-A.

Moreover, after compromising link l_{j^*}, i.e., for $L_m = L_m^{(0)} \cup \{l_{j^*}\}$, the optimization (2) becomes

$$\max_{\widehat{\mathbf{x}}} \sum_{p_i \in P_m^{(0)}} \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) + \sum_{p_i \in P_m^{(1)} \setminus P_m^{(0)}} \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) \tag{8a}$$

$$\text{s.t. } \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) = 0, \qquad \forall p_i \in P_n^{(1)}, \tag{8b}$$

$$\tau_{\max} \ge \widehat{x}_j \ge 0, \qquad \forall l_j \in L_n^{(1)}, \tag{8c}$$

$$\tau \ge \widehat{x}_j \ge 0, \qquad \forall l_j \in L_m^{(1)}, \tag{8d}$$

where $L_m^{(1)}$ ($L_n^{(1)}$) is the new set of compromised (uncompromised) links, and $P_m^{(1)}$ ($P_n^{(1)}$) is the new set of compromised (uncompromised) paths.

Finally, since $P_n^{(1)} \subseteq P_n^{(0)}$, $L_n^{(1)} = L_n^{(0)} \setminus \{l_{j^*}\}$, and $L_m^{(1)} = L_m^{(0)} \cup \{l_{j^*}\}$, any feasible solution to (7) remains feasible for (8). In particular, $\widehat{\mathbf{x}}^{(0)}$ is a feasible solution to (8), with an objective value of $\sum_{p_i \in P_m^{(0)}} \mathbf{r}_i(\widehat{\mathbf{x}}^{(0)} - \mathbf{x}) = \Gamma(L_m^{(0)})$. Thus, under the optimal solution to (8), the objective value $\Gamma(L_m^{(0)} \cup \{l_{j^*}\})$ must be no smaller than $\Gamma(L_m^{(0)})$. □

Step 2 is to show that among all the cuts, L_m^* must be the one that minimizes the total traversal number.

Lemma III.3. *Among all the cuts of P, the optimal set of links to compromise is the cut with the minimum total traversal number.*

Proof. By definition, if L_m is a cut of P, then $P_m = P$ and $P_n = \emptyset$, which simplifies (2) for a given L_m to

$$\max_{\widehat{\mathbf{x}}} \sum_{p_i \in P} \mathbf{r}_i(\widehat{\mathbf{x}} - \mathbf{x}) \tag{9a}$$

$$\text{s.t. } \tau_{\max} \ge \widehat{x}_j \ge 0, \qquad \forall l_j \in L_n, \tag{9b}$$

$$\tau \ge \widehat{x}_j \ge 0, \qquad \forall l_j \in L_m. \tag{9c}$$

It is easy to see that the optimal solution to (9) is $\widehat{x}_j = \tau$ if $l_j \in L_m$ and $\widehat{x}_j = \tau_{\max}$ if $l_j \in L_n$. Under this solution, the objective value of (9) equals

$$\begin{aligned}&\sum_{p_i\in P}(m_i\tau + (|p_i| - m_i)\tau_{\max}) - \sum_{p_i\in P}\mathbf{r}_i\mathbf{x}\\ &= (\tau-\tau_{\max})\sum_{p_i\in P} m_i + \tau_{\max}\sum_{p_i\in P}|p_i| - \sum_{p_i\in P}\mathbf{r}_i\mathbf{x}, \quad (10)\end{aligned}$$

where m_i is the number of compromised links on path p_i and $|p_i|$ is the total number of links on p_i. Only the first term $(\tau - \tau_{\max})\sum_{p_i\in P} m_i$ depends on L_m.

As $\tau - \tau_{\max} \leq 0$, maximizing (10) is equivalent to minimizing $\sum_{p_i\in P} m_i$. We further note that

$$\sum_{p_i\in P} m_i = \sum_{p_i\in P}\sum_{l\in p_i}\mathbb{1}_{l\in L_m} = \sum_{l\in L_m}\sum_{p_i\in P}\mathbb{1}_{l\in p_i} = \sum_{l\in L_m} w_l, \quad (11)$$

where $\mathbb{1}_{\cdot}$ is the indicator function. Thus, the optimal solution to L_m among all the cuts is the cut with the minimum total traversal number. □

Proof of Theorem III.1. By Lemma III.2, L_m^* must be a cut of P. Then by Lemma III.3, it must have the minimum total traversal number among all the cuts. Therefore, L_m^* must be the minimum-traversal cut. □

Remark: The minimum-traversal cut of P may not be unique. From the proof of Theorem III.1, we see that all the minimum-traversal cuts are equally optimal.

Theorem III.1 implies that given a set of targeted paths P, the optimal set L_m of links to compromise is the solution to a novel combinatorial optimization problem as follows.

Definition 2. Given a set of paths P, the *adversarial link selection (ALS)* problem is to find the cut of P with the minimum total traversal number.

C. Hardness Analysis

Below we show the hardness of ALS by connecting it to several well-known hard problems in combinatorial optimization in both the general case and a nontrivial special case.

1) Hardness of General ALS: First, consider the general case of ALS for an arbitrary set of paths P.

Theorem III.4. ALS for an arbitrary path set P is NP-hard.

Proof. To show this, we consider the corresponding decision problem: determine whether a set of paths P has a cut with a given total traversal number T. We will prove that the decision version of ALS is NP-hard by showing a reduction from the *exact cover problem* [48].

Given a set of elements of $E = \{e_1, e_2, \dots, e_n\}$ and a collection S of subsets of E, an exact cover is a subcollection S^* of S such that each element in E is covered once and only once by sets in S^*. To determine if there exists an exact cover is NP-complete [48].

The exact cover problem can be reduced to the following instance of ALS. We construct a set of paths $P = \{p_1, p_2, \dots, p_n\}$ in one-one correspondence with the set of elements $E = \{e_1, e_2, \dots, e_n\}$. Similarly, we construct a set of links $L = \{l_1, l_2, \dots, l_m\}$ in one-one correspondence with the collection of sets $S = \{s_1, s_2, \dots, s_m\}$. The relationship between the paths and the links is such that link l_i is traversed by path p_j if and only if set s_i covers element e_j. Note that such construction is always possible as we allow P to contain arbitrary paths in the general case of ALS. Then we claim that there exists an exact cover S^* of E if and only if the constructed instance of ALS has a cut with a total traversal number of $|P|$.

Suppose that there exists an exact cover S^*, i.e., $E \subseteq \bigcup_{s\in S^*} s$ and $\sum_{s\in S^*}|s| = |E|$. According to the above construction, the corresponding set of links $C^* = \{l_i : s_i \in S^*\}$ must cut each path in P once and only once, and hence C^* is a cut with a total traversal number of $|P|$.

Conversely, suppose that the constructed set of paths P has a cut C^* with a total traversal number of $|P|$. By Definition 1, C^* must cut each path in P once and only once. According to the construction, the corresponding subcollection $S^* = \{s_i : l_i \in C^*\}$ must cover each element in E once and only once, i.e., S^* is an exact cover of E. □

2) Hardness of all-possible-paths ALS: Now consider a special case where P contains all possible paths between a given set K of terminals. This case models networks that employ advanced routing mechanisms such as source routing or Software Defined Networking (SDN), that allow traffic to be routed on any path between a pair of terminals. We call the ALS problem in this special case *all-possible-paths ALS*.

All-possible-paths ALS can reduce to the Multiway Cut problem [49]. Also known as the Multiterminal Cut problem, the Multiway Cut problem is a graph division problem, where given an undirected graph $\mathcal{G}(V, E)$ with link weights $w : E \to \mathbb{R}^+$ and a set of terminals $K \subseteq V$, we want to find a subset of links with the minimum total weight to cut all the paths between the terminals. When the number of terminals equals 2, the Multiway Cut problem becomes the min-cut problem, which can be solved efficiently by the max-flow algorithms. We see that all-possible-paths ALS is a special case of Multiway Cut, where the weights are the traversal numbers. We note that the two problems are not equivalent: in Multiway Cut, the link weights are arbitrary; in all-possible-paths ALS, the link weights are the traversal numbers, which are determined by the network topology and the locations of terminals.

It is known that Multiway Cut is NP-hard, even in a very special case when all the links have unit weights.

Theorem III.5 ([49])**.** The Multiway Cut problem is NP-hard for all $|K| \geq 3$, even if all the link weights are equal to 1.

The hardness of all-possible-paths ALS still remains an open question. Based on Theorem III.5, we conjecture that all-possible-paths ALS is NP-hard, since it is also a special case of Multiway Cut.

Fig. 3 summarizes the relationship between ALS and known NP-hard problems, where the arrows indicate the direction of reduction. As shown in Section III-D1, ALS can reduce to the Weighted Set Cover (WSC) problem, which is also NP-hard.

D. Approximation Algorithms

As ALS is NP-hard, there is no polynomial-time exact algorithm for it unless P = NP. As we mentioned, ALS

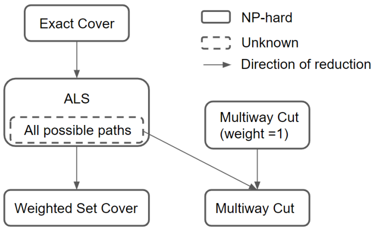


Fig. 3. Relationship between ALS and known NP-hard problems.

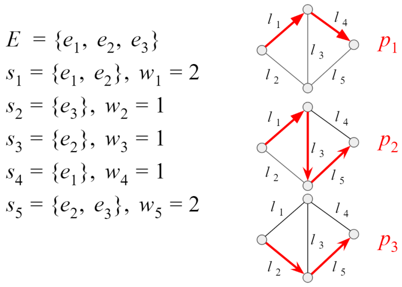


Fig. 4. Reducing ALS to Weighted Set Cover

reduces to Weighted Set Cover (WSC), and all-possible-paths ALS reduces to Multiway Cut. Below we will use known approximation algorithms designed for WSC and Multiway Cut to solve ALS and all-possible-paths ALS, respectively.

1) The greedy algorithm for ALS: Given a set of elements $E = \{e_1, e_2, \ldots, e_n\}$ and a collection $S = \{s_1, s_2, \ldots, s_m\}$ of subsets of E, where each s_i has a weight of w_i, WSC aims at finding the subcollection S^* that covers E with the minimum total weight.

We reduce ALS (for arbitrary P) to WSC as follows. Given a set of paths $P = \{p_1, p_2, \ldots, p_n\}$ traversing a set of links $L = \{l_1, l_2, \ldots, l_m\}$, we construct a set of elements $E = \{e_1, e_2, \ldots, e_n\}$ in one-one correspondence with the paths, and a collection of sets $S = \{s_1, s_2, \ldots, s_m\}$ in one-one correspondence with the links, such that set s_i covers element e_j if and only if link l_i is on path p_j, as illustrated in Fig. 4. Each set s_i has a weight w_i that equals the traversal number of link l_i. It is easy to see that finding the cut with the minimum total traversal number is equivalent to finding the subcollection of sets to cover all the elements with the minimum total weight. We note that in the constructed instance of WSC, the weight of a set always equals its cardinality (i.e., $w_i = |s_i|$), and thus ALS is a special case of WSC.

We apply a well-known greedy algorithm [50], designed for solving WSC, to the ALS problem. The algorithm iterates until all the paths are compromised, where in each iteration, it picks a link with the smallest cost-value ratio and adds the paths traversing it to the set of compromised paths. For a link l, we define the cost-value ratio by $\frac{|P_l|}{|P_l \backslash P_m|}$, where P_l is the set of paths traversing link l. Since the link weight equals $|P_l|$, this ratio is the cost we pay for each newly compromised path, if link l is selected. The pseudocode is shown in Algorithm 1. The while loop (lines 3–6) is repeated $O(|L|)$ times, each iteration taking $O(|L| \cdot |P|)$ time (due to line 4), leading to an overall complexity of $O(|L|^2|P|)$.

Although straightforward, this greedy algorithm is known to have the best approximation guarantee for WSC [50]. Applied to our problem, it guarantees the following.

Theorem III.6 ([50])**.** Algorithm 1 achieves an approximation factor of $H_{|P|} = 1 + \frac{1}{2} + \ldots + \frac{1}{|P|} = \Theta(\log|P|)$ for ALS, i.e., $T^{\text{greedy}} \leq H_{|P|}T^{\text{opt}} = \Theta(log|P|)T^{\text{opt}}$, where T^{greedy} is the total traversal number achieved by Algorithm 1 and T^{opt} is the minimum total traversal number of all the cuts of P.

Algorithm 1: ALS Greedy

```
  input : Paths $P$
  output: Compromised links $L_m$
1 $P_m \leftarrow \emptyset$;
2 $L_m \leftarrow \emptyset$;
3 while $P_m \neq P$ do
4     find the link $l$ with the smallest ratio $\frac{|P_l|}{|P_l \backslash P_m|}$;
5     $P_m \leftarrow P_m \cup P_l$;
6     $L_m \leftarrow L_m \cup \{l\}$;
7 return $L_m$;
```

However, our ultimate goal is to maximize the performance degradation measured by (2a). We can substitute $\sum_{p_i \in P} m_i$ by T^{greedy} in (10) to get the corresponding objective value.

Corollary III.7. Using Algorithm 1 to select the compromised links and the LP (6) to compute the manipulations achieves a total performance degradation of

$$(\tau - \tau_{\max})T^{\text{greedy}} + \tau_{\max} \sum_{p_i \in P} |p_i| - \sum_{p_i \in P} \mathbf{r}_i \mathbf{x} \geq \\ (\tau - \tau_{\max})H_{|P|}T^{\text{opt}} + \tau_{\max} \sum_{p_i \in P} |p_i| - \sum_{p_i \in P} \mathbf{r}_i \mathbf{x}, \quad (12)$$

where T^{greedy} and T^{opt} are defined as in Theorem III.6.

2) CKR relaxation with rounding for all-possible-paths ALS: As mentioned in Section III-C2, all-possible-paths ALS reduces to the Multiway Cut problem, which means we can apply algorithms for Multiway Cut to all-possible-paths ALS.

Calinescu et al. [51] proposed an approach called *CKR relaxation* for Multiway Cut, for which it has been proved that it is NP-hard to achieve a better integrality gap than CKR relaxation for any fixed number of terminals, assuming the Unique Games Conjecture to hold [52]. In a minimization problem, the *integrality gap* is the ratio between the objective value of the optimal integer solution and that of its relaxation, i.e., $\text{OPT}_{\text{int}}/\text{OPT}_{\text{relaxation}}$. We first formulate the Multiway Cut problem as an integer program, and then introduce its CKR relaxation. Given a set V of nodes, a set E of links with weights $(w_{v,v'})_{(v,v') \in E}$, and a set K ($K \subseteq V$) of terminals, the Multiway Cut problem aims at solving

$$\min_{\mathbf{x}} \frac{1}{2} \sum_{(v,v') \in E} \sum_{t \in K} w_{v,v'} |x_{v,t} - x_{v',t}| \quad (13a)$$

$$\text{s.t.} \sum_{t \in K} x_{v,t} = 1, \qquad \forall v \in V, \quad (13b)$$

$$x_{t,t} = 1, \qquad \forall t \in K, \quad (13c)$$

$$x_{v,t} \in \{0, 1\}, \qquad \forall v \in V, t \in K, \quad (13d)$$

where $x_{v,t}$ is the decision variable indicating whether node v will be connected to terminal t after the cut.

By relaxing the integer constraint (13d) and replacing $|x_{v,t} - x_{v',t}|$ by a new variable $y_{v,v',t}$, we get the following:

$$\min_{\mathbf{x},\mathbf{y}} \frac{1}{2} \sum_{(v,v') \in E} \sum_{t \in K} w_{v,v'} y_{v,v',t} \quad (14a)$$

TABLE I
APPROXIMATION ALGORITHMS FOR ALS

algorithm	case	approximation factor
ALS Greedy	general	$\Theta(\log \lvert P\rvert)$
CKR relaxation	all-possible-paths	α^2

$$\text{s.t.} \sum_{t\in K} x_{v,t} = 1, \qquad \forall v \in V, \quad (14b)$$

$$x_{t,t} = 1, \qquad \forall t \in K, \quad (14c)$$

$$x_{v,t} \geq 0, \qquad \forall v \in V,\ t \in K, \quad (14d)$$

$$y_{v,v',t} \geq x_{v,t} - x_{v',t}, \qquad \forall (v,v') \in E,\ t \in K, \quad (14e)$$

$$y_{v,v',t} \geq x_{v',t} - x_{v,t}, \qquad \forall (v,v') \in E,\ t \in K, \quad (14f)$$

which is an LP [50], i.e., an LP relaxation of (13).

For each node v and each terminal t, the solution $\bar{x}_{v,t}$ to the LP relaxation can be viewed as the probability of assigning v to (the connected component containing) t after the cut. A rounding scheme is used to convert this fractional value to either 0 or 1, subject to the constraint (14b). Different rounding schemes lead to different approximation factors. Specifically, the randomized rounding scheme achieves an approximation factor of 1.5 [50], and the best-known rounding scheme can achieve an approximation factor of 1.2965 [53]. Finally, given the rounded value of $x_{v,t}$ ($\forall v \in V,\ t \in K$), the cut is the set of all the links whose endpoints are assigned to different terminals, i.e., $L_m = \{(v,v') \in E :\ \exists t,\ t' \in K \text{ with } t \neq t',\ \text{satisfying } x_{v,t} = x_{v',t'} = 1\}$.

The complexity of this method is dominated by solving the LP (14), which has $O(|K|(|V| + |E|))$ variables and $O(|K|(|V| + |E|))$ constraints, and can be solved in Polynomial($|K|(|V| + |E|)$) time, where the exact order of polynomial depends on the LP algorithm used. For example, the complexity will be $O(|K|^{4.5}(|V| + |E|)^{4.5})$ if using Vaidya's algorithm [54][1].

By similar argument as Corollary III.7, we can bound the overall performance of the attack as follows.

Corollary III.8. Using CKR relaxation with an α-approximation rounding scheme to select the compromised links and the LP (6) to compute the manipulations achieves a total performance degradation of

$$(\tau - \tau_{\max})T^{\text{CKR}} + \tau_{\max} \sum_{p_i \in P} |p_i| - \sum_{p_i \in P} \mathbf{r}_i \mathbf{x} \geq$$
$$(\tau - \tau_{\max})\alpha T^{\text{opt}} + \tau_{\max} \sum_{p_i \in P} |p_i| - \sum_{p_i \in P} \mathbf{r}_i \mathbf{x}, \quad (15)$$

where T^{CKR} is the total traversal number of the links selected by CKR relaxation, and T^{opt} is the minimum total traversal number of all the multiway cuts between the terminals.

TABLE I summarizes the performance guarantee of the aforementioned algorithms in solving ALS.

3) Illustrative example: Consider the example in Fig. 2 (a). ALS Greedy selects the link with smallest cost-value ratio in each iteration (breaking ties arbitrarily), and ends up selecting $L_m = \{l_1, l_3, l_4\}$, as shown in Fig. 5. CKR relaxation first obtains a fractional assignment from each node to the terminals as in Fig. 6 (b), and then rounds it to an integer assignment in Fig. 6 (c) by the scheme in [50]. The output L_m is the set of links in the cut, $L_m = \{l_2, l_4, l_5\}$ in this case.

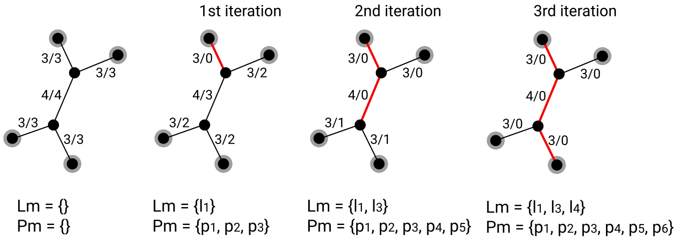


Fig. 5. The illustrative example for ALS Greedy. The network topology is as in Fig. 2 (a), and each link is labeled with its cost-value ratio $\frac{|P_l|}{|P_l \setminus P_m|}$ (randomly breaking ties in 1st and 3rd iterations).

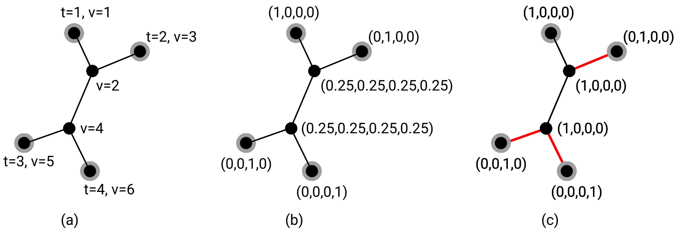


Fig. 6. The illustrative example for CKR relaxation: (a) input topology with node/terminal labels (links labeled as in Fig. 2 (a)); (b) fractional solution to (14) ($(x_{v,t})_{t=1}^{4}$ for each node v); (c) rounded solution (highlighted links are in the cut between the terminals).

IV. PERFORMANCE EVALUATION

We conduct simulations to evaluate the capabilities of an intelligent attacker employing our strategies in comparison with benchmarks, based on real ISP topologies. To be concrete, we consider delay-based DGoS attacks, where the attacker tries to inject the maximum amount of delay onto a set of targeted paths, while the user of these paths tries to localize links with abnormally large delays by network tomography.

A. Experiment Setup

1) Network topology: We use real network topologies from public datasets, whose parameters are shown in the TABLE II. The first four topologies are Point of Presence (PoP)-level topologies from the Internet Topology Zoo [55], and the last two topologies are router-level topologies from the CAIDA project [56]. We classify the topologies into small, medium, and large networks. For each topology, we select a given number of terminals uniformly at random from low-degree nodes (degree ≤ 2), and repeat this selection for 20 times.

2) Parameter setting: For each topology and each set of selected terminals, we compute the paths in P in two ways:

i) *All possible paths*: In this case, P contains all the cycle-free paths between the terminals. Note that cutting all the cycle-free paths is equivalent to cutting all the paths between the terminals. Since the number of all the cycle-free paths can grow exponentially with the network size, we only evaluate this case on the small networks.
ii) *Shortest paths*: In this case, P only contains one shortest path (in hop count) for each pair of terminals, with ties broken arbitrarily. Since there are only $\binom{|K|}{2}$ paths for $|K|$ terminals, we evaluate this case on the medium–large networks.

[1] This algorithm has a worst-case complexity of $O((n+m)^{1.5}nB)$ for an LP with n variables, m constraints, and B input bits.

[2] The constant α depends on the rounding scheme, e.g., 1.5 for randomized rounding and 1.2965 for the rounding scheme in [53].

TABLE II
PARAMETERS OF ISP TOPOLOGIES

Network	size	#nodes	#links	#candidate terminals[4]
Bics	small	33	48	16
BTN	small	53	65	25
Colt	medium	153	191	45
Cogent	medium	197	245	21
AS 20965	large	968	8283	75
AS 8717	large	1778	3755	1075

We assume that before the attack, each link has a delay randomly drawn from $[0, 15]$ ms, and a link is considered "normal" if its delay is within 15 ms, i.e., $\tau = 15$. These parameters are consistent with single-hop delays in real ISP networks [57]. The maximum delay at a link is set to 200 ms, i.e., $\tau_{\max} = 200$, which is within the range of typical buffering capacities at router interfaces[3].

3) Benchmarks: We compare the two proposed algorithms, Algorithm 1 ('ALS greedy') and CKR relaxation with randomized rounding ('CKR'), with the following three heuristics for selecting the set of compromised links:

i) "Random selection" ('random'): To illustrate the capability of an attacker who cannot actively select which links to compromise, this algorithm selects k links uniformly at random, where k is set to the number of compromised links selected by 'CKR' to facilitate comparison.

ii) "Top traversal" ('top traversal'): Based on the intuition that compromising the most traversed links will provide control over more paths, this algorithm selects the k links with the largest traversal numbers. Again, to facilitate comparison, k is set to the number of compromised links selected by 'CKR'.

iii) "Compromise all" ('all'): Compromising all the links is also a very intuitive approach to maximize the damage the attacker can cause to the network.

Under each selection of compromised links, we solve the LP (6) to compute the total performance degradation (measured by the total amount of delay injected by the attacker over all the paths) under the optimal manipulations.

B. Results

Overall, we observe that the proposed algorithms ('ALS greedy' and 'CKR') perform similarly to each other and significantly better than the heuristic algorithms. More importantly, these algorithms show that it is possible to introduce significant delay on communication paths without being localized by network tomography, signaling the need of new defenses.

1) Case of all possible paths: In the case that P contains all the possible paths between the terminals, the results are shown in Fig. 7 (top 2). The y-axis is the performance of the attacker measured by the average injected delay per path (plus/minus one standard deviation), computed over 20 sets of randomly selected terminals, and the x-axis is the number of terminals. In this experiment, 'CKR' performs the best as expected, as it has the best approximation guarantee. Both 'CKR' and 'ALS greedy' perform much better than the heuristic algorithms, demonstrating the importance of carefully selecting the compromised links in modeling the capabilities of intelligent attackers. Among the heuristic algorithms, 'top traversal' performs the best, as it leads to more compromised paths than 'random'. However, it is not sufficient to just compromise more paths. To prevent the compromised links from being detected as bad links by network tomography, the attacker needs to ensure constraint (2e). Therefore, compromising too many links can reduce the attacker's capability in injecting delays. This is why 'all' performs the worst.

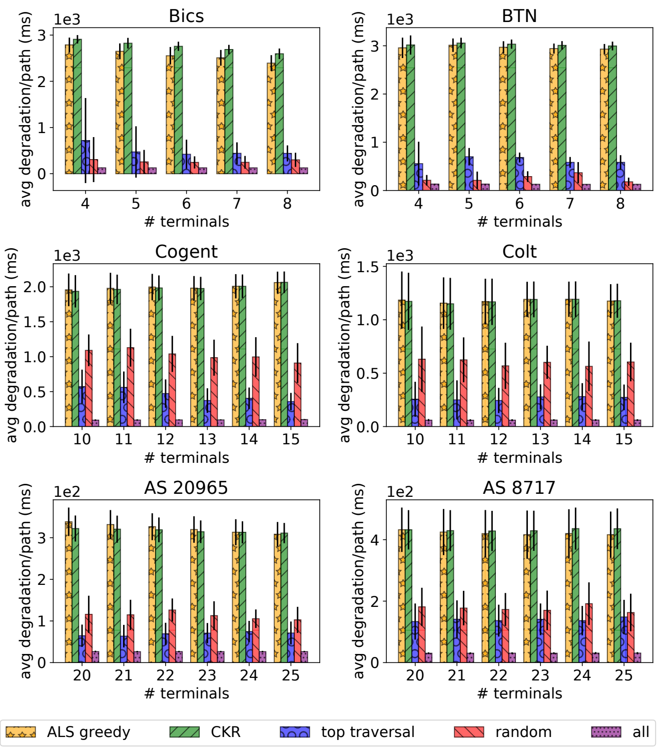


Fig. 7. Average delay degradation for the case of all possible paths (top 2) and the case of shortest paths (bottom 4).

2) Case of shortest paths: Similar results are shown in Fig. 7 (bottom 4) for the case where P only contains the shortest paths between the terminals. We see that 'ALS greedy' and 'CKR' still significantly outperform the other algorithms. However, 'CKR' is not always the best any more, because it is not designed for this case. In particular, 'CKR' will select links to cut all the possible paths between the terminals, while the ALS problem in this case only needs to cut the shortest paths. Because of that, 'CKR' may compromise more links than necessary, which reduces the attacker's capability to manipulate the path delays.

In both cases, the best attack algorithm is able to inject significant delays (0.3–3 seconds/path) without exposing the compromised links to network tomography. Detailed examination of the measurement paths shows that the vulnerability of a network to DGoS attacks is negatively correlated with its identifiability, measured by rank$(R)/|L|$: the average delay degradation per path decreases from 3 seconds to 0.3 second as rank$(R)/|L|$ increases from 0.1 to 0.9. This observation suggests that existing techniques for improving the identifiability via placing monitors and constructing measurement paths [35],

[3] For example, Cisco Supervisor Engine 7-E with a Gigabit port and 32MB memory can buffer traffic for 320 ms [58], and Juniper line cards can buffer traffic for 100–250 ms [59].

[4] For Bics, these are all the nodes with degree ≤ 2; for the other networks, these are all the nodes with degree one.

[36], [37], [38] also help to reduce the vulnerability to DGoS attacks. Note that achieving identifiability does not eliminate this vulnerability, e.g., the paths in Fig. 2 (a) can identify all the links, but DGoS attack can still be launched as in Fig. 2 (c).

V. Constrained Attacks

So far we have assumed that the attacker can compromise any subset of links. In practice, however, there are usually constraints on which and/or how many links the attacker is capable of compromising. To shed light on the impact of such constraints, we will analyze the optimal attack strategy under the constraint that for a given $k > 0$,

$$\sum_{l_j \in L_m} c_j \leq k, \tag{16}$$

where c_j ($c_j \geq 0$) is the cost of compromising link l_j, and k is the total budget of the attacker for compromising links. We use the costs to model the difficulty of controlling the links, e.g., by gaining backdoor access to the associated devices [10], [60] or manipulating the paths (e.g., through BGP hijacking [61]) to position attacker-controlled devices on the links. We assume that these costs can be evaluated by the attacker.

A. Mixed Integer Linear Programming (MILP) Formulation

First of all, we note that the budgeted attack optimization includes the unbudgeted optimization (2) as a special case, which boils down to the ALS problem that is NP-hard as shown in Theorems III.1 and III.4. Thus, the budgeted optimization is also NP-hard. Nevertheless, we will show that this problem can be formulated as an MILP, which allows us to evaluate the maximum damage achievable by a budgeted attacker for small problem instances using MILP solvers.

Specifically, define binary variables

$$\alpha_j := \begin{cases} 1 & \text{if } l_j \in L_m, \\ 0 & \text{otherwise,} \end{cases} \tag{17}$$

$$\beta_i := \begin{cases} 1 & \text{if } p_i \in P_m, \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

Then (2) under the additional constraint (16) can be written as:

$$\max_{\widehat{\mathbf{x}}, \boldsymbol{\alpha}, \boldsymbol{\beta}} \sum_{p_i \in P} \mathbf{r}_i (\widehat{\mathbf{x}} - \mathbf{x}) \tag{19a}$$

$$\text{s.t. } \mathbf{r}_i \widehat{\mathbf{x}} \leq \mathbf{r}_i \mathbf{x} + \beta_i \tau_{\max} \mathbf{r}_i \mathbf{1}, \quad \forall p_i \in P, \tag{19b}$$

$$\mathbf{r}_i \widehat{\mathbf{x}} \geq \mathbf{r}_i \mathbf{x}, \quad \forall p_i \in P, \tag{19c}$$

$$\beta_i \leq \mathbf{r}_i \boldsymbol{\alpha}, \quad \forall p_i \in P, \tag{19d}$$

$$\widehat{x}_j \leq \alpha_j \tau + (1 - \alpha_j) \tau_{\max}, \quad \forall l_j \in L, \tag{19e}$$

$$\widehat{x}_j \geq 0, \quad \forall l_j \in L, \tag{19f}$$

$$\sum_{l_j \in L} \alpha_j c_j \leq k, \tag{19g}$$

$$\alpha_j, \beta_i \in \{0, 1\}, \quad \forall l_j \in L, p_i \in P. \tag{19h}$$

Lemma V.1. The MILP (19) is equivalent to (2) under the additional constraint (16).

Proof. First, we argue that constraints (19b)–(19d) are equivalent to constraints (2b)–(2c). This is because if a path p_i contains no compromised link (i.e., $\mathbf{r}_i \boldsymbol{\alpha} = 0$), then β_i must be zero and thus (19b)–(19c) imply (2b), whereas if p_i contains at least one compromised link, then β_i will be one under the optimal solution, and thus (19b) imposes no constraint on $\widehat{\mathbf{x}}$ (as $\tau_{\max} \mathbf{r}_i \mathbf{1}$ is an upper bound on the path metric for p_i). Moreover, it is easy to see that constraints (19e)–(19f) are equivalent to (2d)–(2e), and constraint (19g) is equivalent to (16). Finally, as uncompromised paths do not incur any degradation, the objective (19a) is equivalent to (2a). □

B. Asymptotic Property of the Optimal L_m

To solve the attack optimization efficiently for large problem instances, we seek to characterize the optimal set of compromised links more explicitly. Generally, adding the budget constraint (16) to (2) will invalidate Theorem III.1. To derive its counterpart under the budget constraint, we have the following result.

Lemma V.2. If $\tau_{\max} \gg \mathbf{r}_i \mathbf{x}$ ($\forall p_i \in P$) and $\tau_{\max} \gg \tau$, then the optimal value of (2) for a given set L_m of compromised links is asymptotically proportional to

$$T_m := \sum_{l_j \in L'_n} \sum_{p_i \in P} r_{ij}, \tag{20}$$

where $L'_n := L_n \setminus \bigcup_{p \in P_n} p$ is the set of uncompromised links that are only traversed by compromised paths.

Proof. We rewrite the objective function (2a) as

$$\sum_{p_i \in P_m} \sum_{l_j \in L} r_{ij} (\widehat{x}_j - x_j) = \sum_{l_j \in L} \sum_{p_i \in P_m} r_{ij} (\widehat{x}_j - x_j). \tag{21}$$

If $l_j \in L_m$, then $\widehat{x}_j \leq \tau$ by (2e). If $l_j \in L_n$, then $\widehat{x}_j \leq \min(\tau_{\max}, \min_{i:\, p_i \in P_n, r_{ij}=1} \mathbf{r}_i \mathbf{x})$ by (2b,2d). For a large $\tau_{\max}$, $\widehat{x}_j$ can achieve $\tau_{\max}$ if and only if $l_j \in L'_n$. Thus, when $\tau_{\max}$ is large, the optimal value of (21) wrt $\widehat{\mathbf{x}}$ is approximately:

$$\tau_{\max} \sum_{l_j \in L'_n} \sum_{p_i \in P_m} r_{ij} = \tau_{\max} \sum_{l_j \in L'_n} \sum_{p_i \in P} r_{ij} \tag{22}$$
$$\propto T_m,$$

where (22) is because the traversal number of compromised paths is equal to the traversal number of measurement paths for $l_j \in L'_n$, i.e., $\sum_{p_i \in P_m} r_{ij} = \sum_{p_i \in P} r_{ij}$ for $l_j \in L'_n$. □

Lemma V.2 immediately yields the following asymptotically equivalent formulation of the budget-constrained DGoS.

Theorem V.3. If $\tau_{\max} \gg \mathbf{r}_i \mathbf{x}$ ($\forall p_i \in P$) and $\tau_{\max} \gg \tau$, then the optimal set L^*_m of compromised links that solves (2) under the additional constraint (16) is the solution to

$$\max T_m \tag{23a}$$

$$\text{s.t. } L_m \subseteq L, \sum_{l_j \in L_m} c_j \leq k, \tag{23b}$$

which we refer to as the *constrained adversarial link selection (CALS)* problem.

Proof. By Lemma V.2, the objective in (2a) is asymptotically equivalent to the objective in (23a) as $\tau_{\max} \to \infty$. □

In words, CALS is a novel combinatorial optimization problem that aims at selecting compromised links subject to a budget constraint to maximize the total traversal number of the uncompromised links that only reside on compromised paths. Similar to ALS, we will show that CALS is also NP-hard.

Corollary V.4. The CALS problem (23) is NP-hard.

Proof. The idea is to show that CALS is actually a generalization of ALS, and hence its NP-hardness is implied by the NP-hardness of ALS as proved in Theorem III.4.

To this end, consider a special case of CALS, where it is known that it suffices to optimize L_m among the cuts of P. If L_m is a cut, then $L'_n = L_n$, and hence $T_m = \sum_{l_j \in L'_n} \sum_{p_i \in P} r_{ij} = \sum_{l_j \in L_n} \sum_{p_i \in P} r_{ij}$, which is the total traversal number of all the uncompromised links. As

$$\sum_{l_j \in L_n} \sum_{p_i \in P} r_{ij} + \sum_{l_j \in L_m} \sum_{p_i \in P} r_{ij} = \sum_{p_i \in P} \sum_{l_j \in L} r_{ij}, \tag{24}$$

which is a constant (the total hop count of all the paths in P), maximizing $\sum_{l_j \in L_n} \sum_{p_i \in P} r_{ij}$ is equivalent to minimizing $\sum_{l_j \in L_m} \sum_{p_i \in P} r_{ij}$ (i.e., minimizing the total traversal number of the compromised links), which is the ALS problem. □

Compared to the MILP formulation (19), although CALS remains NP-hard, it facilitates the development of an efficient suboptimal algorithm as shown below. Moreover, Lemma V.2 implies that T_m can be used as a proxy for analyzing the performance of any strategy for selecting the compromised links.

C. Simple Greedy Algorithm

In the unconstrained case, we have seen in Section IV that the simple greedy algorithm (Algorithm 1) achieves superior performance wrt benchmarks. It is thus natural to consider its extension in the constrained case. The resulting algorithm, called *CALS Greedy*, selects the link that yields the maximum increase in T_m per unit cost in each iteration. Specifically, it follows the same steps as Algorithm 1, except that:

- line 3 is replaced by "**while** $P_m \neq P$ **and** $\exists l \in L \setminus L_m$ such that $L_m \cup \{l\}$ satisfies the budget constraint (16)";
- line 4 is replaced by "find the link l_j with the largest ratio $\frac{T_m(L_m \cup \{l_j\}) - T_m(L_m)}{c_j}$ s.t. $L_m \cup \{l_j\}$ satisfies (16)",

where $T_m(L')$ denotes the value of T_m as defined in (20) when $L_m = L'$. There are again $O(|L|)$ iterations, and each iteration is dominated by the new line 4 above that takes $O(|L|^2|P|)$ time, as we need to evaluate $T_m(L_m \cup \{l_j\})$ for all the $O(|L|)$ candidate links and each evaluation takes $O(|L| \cdot |P|)$ time. The complexity of CALS Greedy is thus $O(|L|^3|P|)$.

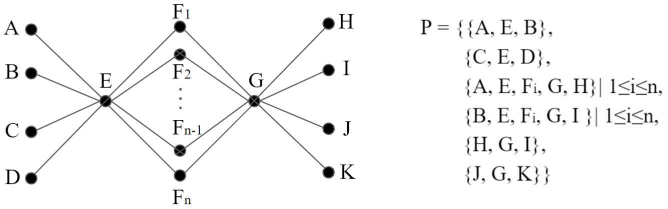


Fig. 8. Example showing the $O(\frac{1}{|P|})$-approximation of CALS Greedy.

Counterexample: As shown later, CALS Greedy performs very well empirically, which raises a question of whether it always closely approximates the optimal solution. To this end, we will show by a counterexample that its approximation factor is at most inversely proportional to the number of paths, i.e., $O(1/|P|)$. Specifically, consider the network topology and the set of paths P as shown in Fig. 8, where $k = 2$ and $c_j = 1$ ($\forall l_j \in L$). CALS Greedy will achieve $T^G_m = 2$ by selecting $L_m = \{(C, E), (K, G)\}$. However, the optimal solution achieves $T^*_m = 6n + 2$ by selecting $L_m = \{(A, E), (G, I)\}$. For this instance of CALS, we see that

$$\frac{T^G_m}{T^*_m} = \frac{1}{3n+1} = \frac{1}{\frac{3}{2}|P| - 5} = \Theta\left(\frac{1}{|P|}\right). \tag{25}$$

Therefore, the approximation factor of CALS Greedy (defined by the worst-case instance) is $O(1/|P|)$. This counterexample shows that the greedy heuristic does not provide a good approximation for CALS in the worst case. Nevertheless, it achieves near-optimal performance for the attacker in average cases as shown in Section V-D. It remains open whether there exists a polynomial-time approximation algorithm for CALS.

Remark: Theoretically, we can also apply the greedy heuristic directly to the original objective (2a). Let $F(L_m)$ denote the optimal objective value of the LP wrt $\widehat{\mathbf{x}}$ under a given L_m, computed as in Section III-A. Then this LP based greedy heuristic will select the link l_j with the maximum $\frac{F(L_m \cup \{l_j\}) - F(L_m)}{c_j}$ in each iteration subject to the budget constraint. This approach, however, will incur a much higher complexity than CALS Greedy due to the need of solving $O(|L|^2)$ LP's. For example, its complexity will be $O(|L|^4(|P|+|L|)^{2.5})$ if using Vaidya's algorithm [54] to solve the LP for $F(L_m)$ (with $O(|L|)$ variables and $O(|P| + |L|)$ constraints).

D. Evaluation

Setup: We evaluate the optimal solution obtained by solving (19) ('MILP') and the proposed algorithm, CALS Greedy ('CALS greedy'), for the budget-constrained DGoS under the setup in Section IV-A, except that we fix the number of terminals and only consider the case of shortest paths. Table III shows the parameter values. As mentioned before, we use costs to model the difficulty for the attacker to control the links, which depends on the specific method of controlling the links and the related parameters, e.g., the models and the vendors of the associated devices for backdoor-based control, or locations of the attacker's devices for hijacking-based control. In our evaluation, the cost of compromising each link is drawn uniformly at random from the interval of $[0, 2)$. This leads to a unit cost per link on the average, giving the budget k an intuitive meaning of the average number of compromised links under random selection. The results are averaged (± one standard deviation) over 20 sets of randomly selected terminals and randomly generated link costs.

Benchmarks: As our problem is a MILP, we use the following heuristics commonly used to solve MILPs as benchmarks:

1) "LP relaxation with Randomized Rounding" ('LP-RR'): This heuristic first solves the LP relaxation of (19), and then treats the fractional solution $(\alpha_j)_{l_j \in L}$ as probabilities for selecting links, subject to the budget constraint.

TABLE III
Parameters for Evaluating Budget-Constrained Attacks

Network	#terminals	budget k	cost
Bics	8	$1, 3, 5, 7, 9, \infty$	[0,2)
BTN	8	$1, 3, 5, 7, 9, \infty$	[0,2)
Cogent	8	$1, 3, 5, 7, 9, \infty$	[0,2)
Colt	10	$1, 3, 5, 7, 9, \infty$	[0,2)
AS 8717	10	$1, 3, 5, 7, 9, \infty$	[0,2)
AS 20965	12	$1, 3, 5, 7, 9, \infty$	[0,2)

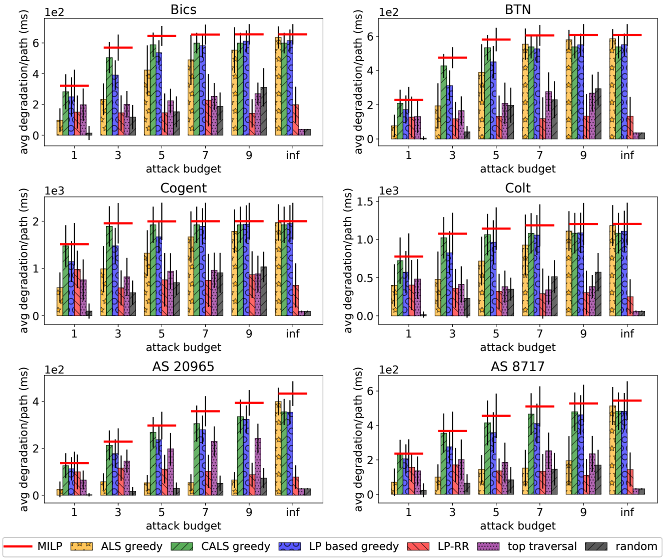


Fig. 9. Average delay degradation under the budget constraint.

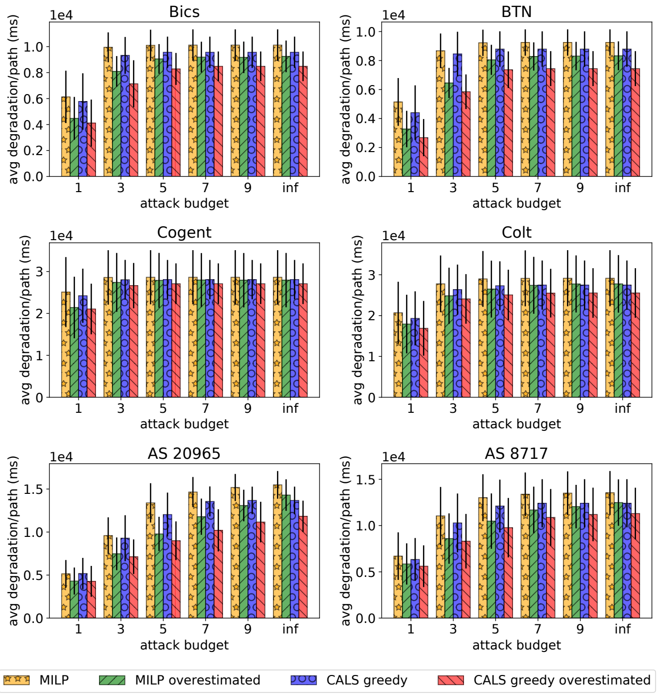


Fig. 10. Average delay degradation when the attacker overestimates the measurement paths ($|\tilde{P}|/|P| = 50\%$).

2) “LP based greedy”: This heuristic directly maximizes (2a) by selecting one more compromised link per iteration that yields the maximum additional delay degradation per unit cost, subject to the budget constraint.

We also adapt algorithms from the unconstrained case as benchmarks. Algorithm 1 (‘ALS greedy’) can be easily adapted to satisfy the budget constraint (16) by stopping after exhausting the budget. Similarly, the heuristics ‘random’ and ‘top traversal’ (see Section IV-A3) can also be easily adapted to satisfy the budget constraint (16).

Results: Fig. 9 shows the comparison of different attack strategies, where $k = \infty$ is the unconstrained case. *First of all*, the result shows that it is necessary to change the objective from minimizing the total traversal number to maximizing T_m (20) when we (the attacker) are not able to compromise a cut due to the budget constraint. This is indicated by the poor performance of ‘ALS greedy’ when k is small. *Secondly,* the proposed algorithm, ‘CALS greedy’, achieves as much damage as ‘ALS greedy’ under an unlimited budget, but much more damage under a limited budget. Across all budget values, ‘CALS greedy’ achieves near-optimal performance (i.e., close to ‘MILP’). *Finally,* ‘CALS greedy’ outperforms the other heuristics derived from our optimization formulation (i.e., ‘LP-RR’, ‘LP based greedy’). In particular, while ‘LP based greedy’ achieves comparable performance degradation, it is much slower, e.g., the experiment on AS 20965 takes 17 seconds for ‘CALS greedy’ but 4844 seconds for ‘LP based greedy’, indicating the value of using T_m (20) as a proxy objective function. Overall, we see that even though ‘CALS greedy’ does not guarantee a good approximation in the worst case (see the counterexample in Section V-C), it performs well in average cases.

Moreover, we evaluate the impact of imperfect knowledge of measurement paths, by setting the attacker-assumed measurement paths P to the set of all the shortest paths between the terminals and the actual measurement paths $\tilde{P}$ used by network tomography to a random subset of these paths. We then repeat the experiment in Fig. 9, except that we only count the performance degradation on the paths in $\tilde{P}$ (assuming that only these paths are used). Fig. 10 shows the average degradations achieved by the optimal attack strategy (‘MILP’) and the best polynomial-time strategy (‘CALS greedy’) with accurate knowledge of the measurement paths, together with those without accurate knowledge (‘overestimated’). The results show that although the attacker incurs some suboptimality due to not knowing the exact set of communication paths, he can still launch the attack on all possible paths and achieve significant damage on the actually used paths. Here we have assumed that network tomography can only monitor the paths used for communications; our recent work [62] considers a more general scenario where network tomography can monitor additional paths via active probing, in which case only the performance degradation on data communication paths matters.

Finally, as our attack model does not rule out the possibility for a specific inference algorithm to detect some of the compromised links as bad links, we evaluate the detectability of compromised links under our best polynomial-time attack strategy, CALS Greedy. To this end, we implement a tomography-based detector that tries to explain all the measurements with the minimum number of bad links as in [18]. Let L_d denote the set of detected bad links. Fig. 11 shows the fraction of detected compromised links $|L_d \cap L_m|/|L_m|$ (‘detection rate’), the fraction of detected uncompromised links $|L_d \cap L_n|/|L_n|$ (‘false alarm rate’), the

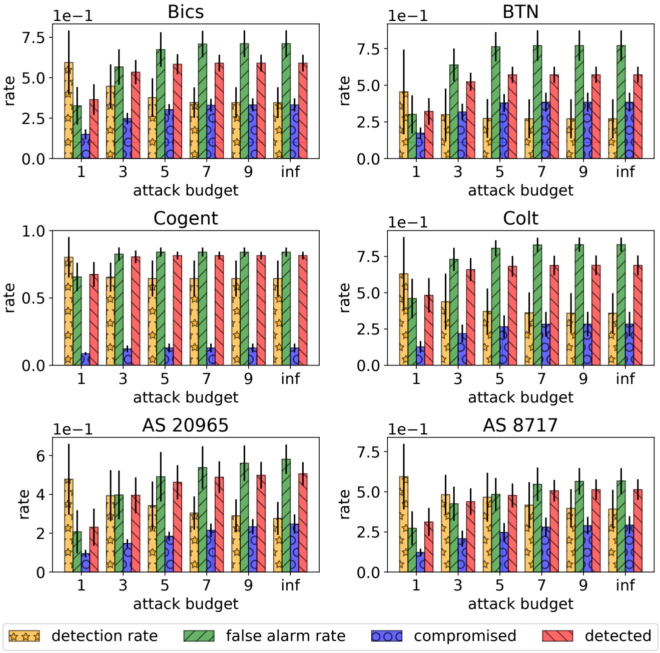


Fig. 11. Performance of tomography-based bad link detection.

fraction of compromised links $|L_m|/|L|$ ('compromised'), and the fraction of detected links $|L_d|/|L|$ ('detected'). We see that although the detector detects some compromised links as bad links, it detects even more innocent links as bad links, and is thus unable to localize the compromised links. Specifically, the false alarm rate is comparable to or even higher than the detection rate in most attack scenarios, meaning that an uncompromised link is more likely to be flagged as a bad link than a compromised link. This is because the detector flags a much larger number of bad links than the actual number of links introducing significant delays (i.e., compromised links). For example, in Cogent, the detector claims that 70–80% links are bad while most of the delays are injected by 10% of the links. This indicates the difficulty of localizing the compromised links using existing tomography techniques.

VI. Conclusion

This work helps to establish the fundamental limit of network tomography in adversarial environments by formulating and analyzing a novel type of attack, called the stealthy DeGrading of Service (DGoS) attack, that aims at maximally degrading the performance of targeted paths without being localized by network tomography. Through careful analysis, we derive explicit properties of the optimal attack strategy. The derived properties allow us to link our problem to well-known combinatorial optimization problems, and leverage existing algorithms with approximation guarantees. Our evaluations on real topologies show that the proposed attack can significantly degrade communication performances without being localized by network tomography, signaling the need of new defenses. In particular, our evaluations show a notable performance gap between heuristic attack strategies (e.g., compromising the most traversed links) and the proposed strategies, demonstrating the importance of modeling intelligent attackers.

Discussion: Our results suggest several potential approaches to the defense. First, our proposed algorithms can be used to identify the set of links that once controlled by an adversary, will cause the maximum damage, which helps to select links for protection (e.g., by installing monitoring agents or updating software/hardware at the endpoints). Moreover, the results in Fig. 10 suggest that dynamically adapting the measurement paths among a larger set of paths can help network tomography to mitigate DGoS attacks by making it harder for the attacker to learn which paths are monitored. Lastly, our observations that networks with higher identifiability are less vulnerable to DGoS attacks suggest that existing measurement design algorithms can also help to defend against such attacks. We leave detailed investigation of these defenses to future work.

References

[1] C. Chiu and T. He, "Stealthy DGoS attack: Degrading of service under the watch of network tomography," in *IEEE INFOCOM*, July 2020.

[2] A. Dhamdhere, R. Teixeira, C. Dovrolis, and C. Diot, "Netdiagnoser: Troubleshooting network unreachabilities using end-to-end probes and routing data," in *ACM CoNEXT*, 2007.

[3] S. Zarifzadeh, M. Gowdagere, and C. Dovrolis, "Range tomgoraphy: Combining the practicality of boolean tomography with the resolution of analog tomography," in *ACM IMC*, 2012.

[4] D. Ghita, K. Argyraki, and P. Thiran, "Toward accurate and practical network tomography," *ACM SIGOPS Operating Systems Review*, vol. 47, no. 1, pp. 22–26, January 2013.

[5] N. Harvey, M. Patrascu, Y. Wen, S. Yekhanin, and V. Chan, "Non-adapative fault diagnosis for all-optical networks via combinatorial group testing on graphs," in *IEEE INFOCOM*, 2007.

[6] S. Ahuja, S. Ramasubramanian, and M. Krunz, "SRLG failure localization in optical networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 4, pp. 989–999, Auguest 2011.

[7] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, "Network tomography: Recent developments," *Statistical Science*, 2004.

[8] Y. Zhao, Y. Chen, and D. Bindel, "Towards unbiased end-to-end network diagnosis," in *ACM SIGCOMM*, 2006.

[9] C. A. Shue, A. J. Kalafut, and M. Gupta, "Abnormally macilious autonomous systems and their Internet connectivity," *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, pp. 220–230, February 2012.

[10] L. Constantin, "Attackers slip rogue, backdoored firmware onto Cisco routers," PC World - Security, 2015.

[11] Z. Zhang, O. Mara, and K. Argyraki, "Network neutrality inference," in *ACM SIGCOMM*, 2014.

[12] S. Zhao, Z. Lu, and C. Wang, "When seeing isn't believing: On feasibility and detectability of scapegoating in network tomography," in *IEEE ICDCS*, 2017.

[13] Y. Vardi, "Estimating source-destination traffic intensities from link data," *Journal of the American Statistical Assoc.*, pp. 365–377, 1996.

[14] M. Coates, A. O. Hero, R. Nowak, and B. Yu, "Internet tomography," *IEEE Signal Processing Magzine*, vol. 19, pp. 47–65, 2002.

[15] M. F. Shih and A. O. Hero, "Unicast inference of network link delay distributions from edge measurements," in *IEEE ICASSP*, 2001.

[16] V. N. Padmanabhan, L. Qiu, and H. Wang, "Server-based inference of internet link lossiness," in *IEEE INFOCOM*, April 2003.

[17] N. Duffield, "Simple network performance tomography," in *ACM SIGCOMM conference on Internet measurement*, 2003.

[18] ——, "Network tomography of binary network performance characteristics," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5373–5388, December 2006.

[19] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley, "Multicase-based inference of network internal loss characteristics," *IEEE Transactions on Information Theory*, vol. 45, no. 7, pp. 2462–2480, November 1999.

[20] A. Adams, T. Bu, T. Friedman, J. Horowitz, D. Towsley, R. Caceres, N. Duffield, F. Presti, and V. Paxson, "The use of end-to-end multicast measurements for characterizing internal network behavior," *IEEE Communications Magazine*, vol. 38, no. 5, pp. 152–159, May 2000.

[21] N. Duffield and F. Lo Presti, "Multicast inference of packet delay variance at interior network links," in *IEEE INFOCOM*, 2000.

[22] F. Lo Presti, N. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal delay distributions," *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 761–775, Dec. 2002.

[23] Y. Xia and D. Tse, "Inference of link delay in communication networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2235–2248, December 2006.

[24] N. Duffield, F. LoPresti, V. Paxson, and D. Towsley, "Network loss tomography using striped unicast probes," *IEEE/ACM Transactions on Networking*, vol. 14, no. 4, pp. 697–710, August 2006.

[25] B. Xi, G. Michailidis, and V. Nair, “Estimating network loss rates using active tomography,” *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1430–1448, December 2006.

[26] E. Lawrence, G. Michailidis, and V. N. Nair, “Network delay tomography using flexicast experiments,” *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, vol. 68, no. 5, pp. 785–813, 2006.

[27] M. Coates and R. Nowak, “Network tomography for internal delay estimation,” in *IEEE ICASSP*, May 2001.

[28] M. F. Shih and A. O. Hero, “Unicast-based inference of network link delay distributions using mixed finite mixture models,” *IEEE Transactions on Signal Processing, Special Issue on Signal Processing in Networking*, vol. 51, no. 9, pp. 2219–2228, August 2003.

[29] O. Gurewitz and M. Sidi, “Estimating one-way delays from cyclic-path delay measurements,” in *IEEE INFOCOM*, 2001.

[30] Y. Chen, D. Bindel, and R. H. Katz, “An algebraic approach to practical and scalable overlay network monitoring,” in *ACM SIGCOMM*, 2004.

[31] A. Chen, J. Cao, and T. Bu, “Network tomography: Identifiability and Fourier domain estimation,” in *IEEE INFOCOM*, 2007.

[32] H. Nguyen and P. Thiran, “The Boolean solution to the congested IP link location problem: Theory and practice,” in *IEEE INFOCOM*, 2007.

[33] Q. Zheng and G. Cao, “Minimizing probing cost and achieving identifiability in probe-based network link monitoring,” *IEEE Transactions on Computers*, vol. 62, no. 3, pp. 510–523, March 2013.

[34] S. Tati, S. Silvestri, T. He, and T. LaPorta, “Robust network tomography in the presence of failures,” in *IEEE ICDCS*, 2014.

[35] A. Gopalan and S. Ramasubramanian, “On identifying additive link metrics using linearly independent cycles and paths,” *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, 2012.

[36] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, “Identifiability of link metrics based on end-to-end path measurements,” in *ACM IMC*, 2013.

[37] ——, “Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement,” *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1351–1368, June 2014.

[38] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, “Efficient identification of additive link metrics via network tomography,” in *IEEE ICDCS*, 2013.

[39] S. Ahuja, S. Ramasubramanian, and M. Krunz, “SRLG failure localization in all-optical networks using monitoring cycles and paths,” in *IEEE INFOCOM*, 2008.

[40] S. Cho and S. Ramasubramanian, “Localizing link failures in all-optical networks using monitoring tours,” *Elsevier Computer Networks*, vol. 58, pp. 2–12, January 2014.

[41] L. Ma, T. He, A. Swami, D. Towsley, K. Leung, and J. Lowe, “Node failure localization via network tomography,” in *ACM IMC*, 2014.

[42] L. Ma, T. He, A. Swami, D. Towsley, and K. Leung, “On optimal monitor placement for localizing node failures via network tomography,” *Elsevier Performance Evaluation*, vol. 91, pp. 16–37, September 2015.

[43] ——, “Network capability in localizing node failures via end-to-end path measurements,” *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 434–450, February 2017.

[44] Y. Chen, D. Bindel, H. Song, and R. Katz, “Algebra-based scalable overlay network monitoring: Algorithms, evaluation, and applications,” *IEEE/ACM Transactions on Networking*, vol. 15, pp. 1084–1097, October 2007.

[45] Q. Liang and E. Modiano, “Optimal network control with adversarial uncontrollable nodes,” in *ACM Mobihoc*, 2019.

[46] “Threshold based alerting on NetFlow.” [Online]. Available: https://www.manageengine.com/products/netflow/threshold-based-alerts.html

[47] “ManageEngine OpManager.” [Online]. Available: https://www.manageengine.com/network-monitoring/?nfa

[48] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.

[49] E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis, “The complexity of multiterminal cuts,” *SIAM Journal on Computing(SICOMP)*, vol. 23, 1994.

[50] V. V. Vazirani, *Approximation Algorithm*. Springer, 2001.

[51] G. Călinescu, H. Karloff, and Y. Rabani, “An improved approximation algorithm for multiway cut,” *Computer and System Sciences*, vol. 60, pp. 564–574, June 2000.

[52] R. Manokaran, J. S. Naor, P. Raghavendra, and R. Schwartz, “Sdp gaps and ugc hardness for multiway cut, 0-extension and metric labeling,” in *ACM STOC*, 2008.

[53] A. Sharma and J. Vondrák, “Multiway cut, pairwise realizable distributions, and descending thresholds,” in *ACM STOC*, 2014.

[54] P. M. Vaidya, “Speeding-up linear programming using fast matrix multiplication,” in *IEEE FOCS*, 1989.

[55] “The Internet Topology Zoo,” http://www.topology-zoo.org/dataset.html.

[56] “Center for Applied Internet Data Analysis: Macroscopic Internet Topology Data Kit (ITDK),” http://www.caida.org/data/internet-topology-data-kit/.

[57] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, and C. Dio, “Measurement and analysis of single-hop delay on an IP backbone network,” *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 908–921, August 2003.

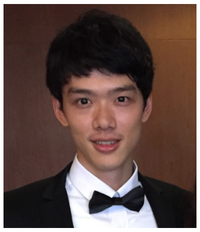
[58] “cat4510 buffer size.” [Online]. Available: https://community.cisco.com/t5/switching/cat4510-buffer-size/td-p/2872535

[59] “Buffer size ex9200 line card.” [Online]. Available: https://forums.juniper.net/t5/Ethernet-Switching/Buffer-size-Ex9200-line-card/td-p/477262

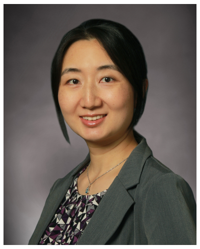
[60] A. Greenberg, “Stealthy, destructive malware infects half a million routers,” Wired - Security, 2018.

[61] Z. Julian, “An overview of BGP hijacking,” Bishop Fox, 2015.

[62] C. Chiu and T. He, “Stealthy DGoS attack under passive and active measurements,” in *IEEE Globecom*, December 2020.



Cho-Chun Chiu (S’20) received the B.S. degree in Space Science and Engineering from National Central University in 2009 and M.S. in Mechanical Engineering from National Taiwan University in 2011. He is a Ph.D. student in Computer Science and Engineering at the Pennsylvania State University, advised by Prof. Ting He. His research interest includes computer networking, network security, differential privacy, and federated learning.



Ting He (SM’13) received the B.S. degree in computer science from Peking University, China, in 2003 and the Ph.D. degree in electrical and computer engineering from Cornell University, Ithaca, NY, in 2007. Dr. He is an Associate Professor in the School of Electrical Engineering and Computer Science at Pennsylvania State University, University Park, PA. Her work is in the broad areas of computer networking, network modeling and optimization, and statistical inference. Dr. He is a senior member of IEEE, an Associate Editor for IEEE Transactions on Communications (2017-2020) and IEEE/ACM Transactions on Networking (2017-2021), and an Area TPC Chair of IEEE INFOCOM (2021). She received multiple Outstanding Contributor Awards from IBM, and multiple paper awards from ITA, ICDCS, SIGMETRICS, and ICASSP.