This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

REACT to Cyber Attacks on Power Grids

Saleh Soltan, *Member, IEEE,* Mihalis Yannakakis, and Gil Zussman, *Senior Member, IEEE*

Abstract—Motivated by the recent cyber attack on the Ukrainian power grid, we study cyber attacks on power grids that affect both the physical infrastructure and the data at the control center–which therefore are cyber-physical in nature. In particular, we assume that an adversary attacks an area by: (i) remotely disconnecting some lines within the attacked area, and (ii) modifying the information received from the attacked area to mask the line failures and hide the attacked area from the control center. For the latter, we consider two types of attacks: (i) *data distortion:* which distorts the data by adding powerful noise to the actual data, and (ii) *data replay:* which replays a locally consistent old data instead of the actual data. We use the DC power flow model and prove that the problem of finding the set of line failures given the phase angles of the nodes outside of the attacked area is strongly NP-hard, even when the attacked area is known. However, we introduce the polynomial time REcurrent Attack Containment and deTection (REACT) Algorithm to approximately detect the attacked area and line failures after a cyber-physical attack. We numerically show that it performs well in detecting the attacked area, and detecting single, double, and triple line failures in small and large attacked areas.

Index Terms—Power Grids; Cyber-Physical attacks; False Data Injection; Line Failures Detection; Graph Theory; Algorithms

1 Introduction

Due to their complexity and magnitude, modern infrastructure networks need to be monitored and controlled using computer systems. These computer systems are vulnerable to cyber attacks [1]. One of the most important infrastructure networks that is vulnerable to cyber attacks is the power grid which is monitored and controlled by the Supervisory Control And Data Acquisition (SCADA) system.

In a recent cyber attack on the Ukrainian power grid [2], the attackers stole credentials for accessing the SCADA system and used them to cause a large scale blackout affecting hundred thousands of people. In particular, they simultaneously operated several of the circuit breakers in the grid and jammed the phone lines to keep the system operators unaware [2].

Motivated by the Ukraine event, in this paper, we deploy the DC power flow model and study a model of a cyber-physical attack on the power grid that affects both the physical infrastructure and the data at the control center. We assume that an adversary attacks an area by: (i) disconnecting some lines within the attacked area (by remotely activating the circuit breakers), and (ii) modifying the information (phase angles of the nodes and status of the lines) received from the attacked area to mask the line failures and hide the attacked area from the control center. For the latter, we consider two types of attacks: (i) *data distortion:* which distorts the data by adding powerful noise to the data received from the attacked area, and (ii) *data replay:* which replays a locally consistent old data instead of the actual data. We assume that the system reaches a steady-state after the attack. Fig. 1 shows an example of such an attack.

We prove that the problem of finding the set of line failures given the phase angles of the nodes outside of the attacked area is strongly NP-hard, *even when the attacked area is known*. Hence, one cannot expect to develop a polynomial time algorithm that can exactly detect the attacked area and recover the information for all possible attack scenarios. However, we introduce the polynomial time REcurrent Attack Containment and deTection (REACT) Algorithm and numerically show that it performs very well in reasonable scenarios.

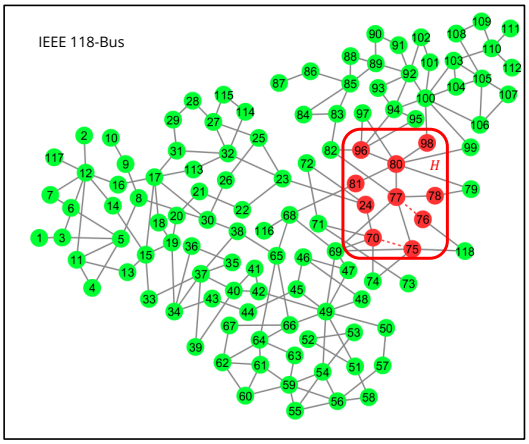


Fig. 1: The attack model. An adversary attacks an area H which is unknown to the control center (represented by red nodes) by disconnecting some lines within the attacked area (shown by red dashed lines) and modifying the information received from the attacked area to mask the line failures and hide the attacked area from the control center.

In particular, we first introduce the ATtacked Area Containment (ATAC) Module for approximately detecting the attacked area using graph theory and the algebraic properties of the DC power flow equations. We show that the ATAC Module can always provide an area containing the attacked area after a data distortion or a data replay attack. We further provide tools to improve the accuracy of the approximated attacked areas obtained by the ATAC Module.

Then, we introduce the randomized LIne Failures De-

S. Soltan is with the Elec. Eng. Dept. at Princeton University, Princeton, NJ (e-mail: ssoltan@princeton.edu). This work was done while Saleh Soltan was with Columbia University. G. Zussman is with the Elec. Eng. Dept. at Columbia University, New York, NY (e-mail: gil@ee.columbia.edu) and Mihalis Yannakakis is with the Comp. Sci. Dept. (email: mihalis@cs.columbia.edu) at Columbia University, New York, NY.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

tection (LIFD) Module to detect the line failures and recover the phase angles inside the detected attacked area. The LIFD Module builds upon the methods first introduced in [3], to detect line failures using Linear Programming (LP) in more general cases. In particular, we prove that in some cases that the methods in [3] fail to detect line failures, the LIFD Module can successfully detect line failures in expected polynomial running time.

Finally, the REACT Algorithm combines the ATAC and LIFD Modules to provide a comprehensive algorithm for attacked area detection and information recovery following a cyber-physical attack. We evaluate the performance of the REACT Algorithm by considering two attacked areas, one with 15 nodes and the other one with 31 nodes within the IEEE 300-bus system [4]. We show that when the attacked area is small, the REACT Algorithm performs equally well after the data distortion and the data replay attacks. In particular, it can exactly detect the attacked area in all the cases, and accurately detect single, double, and triple line failures within the attacked area in more than 80% of the cases.

When the attacked area is large, however, the REACT Algorithm's performance is different after the data distortion and the data replay attacks. It still performs very well in detecting the attacked area after a data distortion attack and accurately detects line failures after single, double, and triple line failures in more than 60% of the cases. However, it may face difficulties providing an accurate approximation of the attacked area after a replay attack. Despite these difficulties in approximating the attacked area, it accurately detects single and double line failures in around 98% and 60% of the cases, respectively.

The main contributions of this paper are: (i) analyzing the computational complexity of the attacked area detection and information recovery problem after a cyber-physical attack on the grid, (ii) introducing a module to detect the attacked area after such an attack, and (iii) introducing a randomized weight linear program for detecting line failures in the large attacked areas in expected polynomial time.

2 Related Work

Attacks on general networks was thoroughly studied in the past (e.g., [5]–[9] and references therein). In particular, [10], [11] studied a problem similar to the one studied in this paper (failure detection from partial observations) in the context of communication networks. However, due to fundamental differences between power flows and data flows, these works are not extendable to power systems.

Power systems' vulnerability to failures and uncertainties was also widely studied in the past few years [12]–[20]. In particular false data injection attacks on power grids and anomaly detection were studied using the DC power flows in [21]–[27]. These studies focused on the observability of the failures and attacks in the grid. In the related problem of Bad Data Detection (BDD) in the SCADA system [28], the objective is to detect the bad data injected by the attacker when the attack has no physical components. Hence, the existing methods for BDD cannot be used in the scenarios studied in this paper for detecting line failures.

The problem of line failures detection using phase angle measurements during the normal operation of the grid were studied in [29]–[31]. The problem of line failures detection in an area based on the information from external nodes was first studied in [32] using sparse recovery methods. In [3], attack scenarios similar to the one in this paper was studied. However, [3] only focused on the attacks that blocked the information from the attacked area, and therefore, the attacked area was detectable simply by checking the missing data. Moreover, the line failures method provided in [3] was limited to certain topologies for the attacked area. In recent works [33], [34], the methods in [3] were extended to function under the AC power flow model. Similar to [3], problems in [33], [34] are focused on the attacks that block the information from the attacked area. Hence, in these works, detecting the attacked area is straight forward. Moreover, the techniques in [3], [33], [34] fail to detect *all* the line failures as the attacked area becomes larger, but the LIFD Module presented in this paper, uses randomization to detect all the line failures in large attacked areas as well.

Finally, in a recent series of works, the vulnerability of power grids to undetectable cyber-physical attacks is studied [35]–[37] using the DC power flows. These studies consider different scenarios in terms of available information and are mainly focused on designing attacks that affect the entire grid and therefore may be impossible to detect.

3 Model and Definitions

3.1 DC Power Flow Model

In this work, we focus on the power systems' transmission network. Hence, the term "power grid" mainly denotes the transmission network. We use the linearized DC power flow model, which is widely used as an approximation for the non-linear AC power flow model in studying vulnerabilities of power grids [3], [18], [35]–[37]. The notation is summarized in Table 1. In particular, we represent the power grid by a connected undirected graph $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$ and $E = \{e_1, \ldots, e_m\}$ are the set of nodes and edges corresponding to the buses and transmission lines, respectively. Each edge e_i is a set of two nodes $e_i = \{u, v\}$. p_v is the active power *supply* ($p_v > 0$) or *demand* ($p_v < 0$) at node $v \in V$ (for a *neutral node* $p_v = 0$). We assume *pure reactive* lines, implying that each edge $\{u, v\} \in E$ is characterized by its *reactance* $r_{uv} = r_{vu}$.

Given the power supply/demand vector $\vec{p} \in \mathbb{R}^{|V|\times 1}$ and the reactance values, a *power flow* is a solution $\mathbf{P} \in \mathbb{R}^{|V|\times|V|}$ and $\vec{\theta} \in \mathbb{R}^{|V|\times 1}$ of:

$$\sum_{v\in N(u)} p_{uv} = p_u, \ \forall\, u \in V \tag{1}$$

$$\theta_u - \theta_v - r_{uv}p_{uv} = 0, \ \forall\, \{u, v\} \in E \tag{2}$$

where $N(u)$ is the set of neighbors of node u, p_{uv} is the power flow from node u to node v, and θ_u is the phase angle of node u. Eq. (1) guarantees (classical) flow conservation and (2) captures the dependency of the flow on the reactance values and phase angles. Additionally, (2) implies that $p_{uv} = -p_{vu}$. When the total supply equals the total demand in each connected component of G, (1)-(2) has a unique solution $\mathbf{P}$ and $\vec{\theta}$ up to a shift (since shifting all

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

TABLE 1: Summary of notation.

Notation	Description
$G = (V, E)$	The graph representing the power grid
$\mathbf{A}$	Admittance matrix of G
$\vec{\theta}$	Vector of the phase angles of the nodes in G
$\vec{p}$	Vector of power supply/demand values
H	A subgraph of G representing the attacked area
F	Set of failed edges due to an attack
$\mathbf{D}$	Incidence matrix of G
$N(i)$	Set of neighbors of node i
$N(S)$	Set of neighbors of subgraph S
$\mathrm{int}(S)$	Interior of the subgraph S
$\partial(S)$	Boundary of the subgraph S
$\mathrm{cl}(S)$	Closure of the subgraph S
$\bigcirc'$	The actual value of $\bigcirc$ after an attack
$\bigcirc^\star$	The observed value of $\bigcirc$ after an attack
$\overline{\bigcirc}$	The complement of $\bigcirc$

θ_us by equal amounts does not violate (2)). Eqs.(1)-(2) are equivalent to the following matrix equation:

$$\mathbf{A}\vec{\theta} = \vec{p} \tag{3}$$

where $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$ is the *admittance matrix* of G,[1] defined as:

$$a_{uv} = \begin{cases} 0 & \text{if } u \neq v \text{ and } \{u,v\} \notin E, \\ -1/r_{uv} & \text{if } u \neq v \text{ and } \{u,v\} \in E, \\ -\sum_{w\in N(u)} a_{uw} & \text{if } u = v. \end{cases}$$

Note that in power grids nodes can be connected by multiple edges, and therefore, if there are k multiple lines between nodes u and v, $a_{uv} = -\sum_{i=1}^{k} 1/r_{uv_i}$. Once $\vec{\theta}$ is computed, the flows, p_{uv}, can be obtained from (2).

Notation. Throughout this paper we use bold uppercase characters to denote matrices (e.g., $\mathbf{A}$), italic uppercase characters to denote sets (e.g., V), and italic lowercase characters and overline arrow to denote column vectors (e.g., $\vec{\theta}$). For a matrix $\mathbf{Q}$, $\mathbf{Q}_i$ denotes its i^{th} row, and q_{ij} denotes its $(i,j)^{\text{th}}$ entry. For a column vector $\vec{y}$, $\vec{y}^T$ denote its transpose, y_i denotes its i^{th} entry, $\|\vec{y}\|_1 := \sum_{i=1}^{n} |y_i|$ is its l_1-norm, and $\mathrm{supp}(\vec{y}) := \{i | y_i \neq 0\}$ is its support.

3.2 Attack Model

We study cyber attacks on the power grid that affect both grid's physical infrastructure and the data at its control center–which therefore are cyber-physical in nature. We assume that an adversary attacks an area by: (i) disconnecting some lines within the attacked area (by remotely activating the circuit breakers), and (ii) modifying the information (phase angle of the nodes and status of the lines) received from the attacked area to mask the line failures and hide the attacked area from the control center. We assume that the system reaches a steady-state after the attack. Hence, supply/demand values do not change after the attack and disconnecting lines within the attacked area does not make G disconnected. However, the developed methods in this paper can also be used when these conditions do not hold, if the control center is aware of the changes in the supply/demand values after the attack and in the case of the grid separation. We also assume that system operator has a complete knowledge of the state of the system before the attack, namely $\mathbf{A}$, $\vec{\theta}$, and $\vec{p}$.

An attacked area is an induced subgraph of G like $H = (V_H, E_H)$. Fig. 1 depicts an example of such an attack on the attacked area represented by H. Due to the attack, some lines within the attacked area (i.e., in E_H) are disconnected (we refer to these edges as *failed lines*), and the reported phase angles and the status of the lines from within the *attacked area* are modified. We denote the set of failed lines in area H by $F \subseteq E_H$. Upon failure, the failed lines are removed from the graph and the flows are redistributed according to (1)-(2). The objective is to detect the attacked area H and the failed lines F after the attack using the observed modified phase angles. Notice that the attacked area represents the induced subgraph by a set of nodes for which the measurements are manipulated by the attacker. Hence, the scenario that the attacker manipulates the measurements in larger area than the area for which he can disconnect the lines, is a special case of the scenarios studied here.

The vectors of phase angle of the nodes in H and in its complement $\bar{H} = G\backslash H$ are denoted by $\vec{\theta}_H$ and $\vec{\theta}_{\bar{H}}$, respectively. We use the prime symbol ($'$) to denote the actual values after an attack. For instance, G', $\mathbf{A}'$, and $\vec{\theta}'$ are used to represent the graph, the admittance matrix of the graph, and the actual phase angles after the attack. Based on our assumptions $\vec{p} = \mathbf{A}\vec{\theta} = \mathbf{A}'\vec{\theta}' = \vec{p}'$.

We also use $\vec{\theta}^\star$ to denote the observed phase angles after the attack. According to the attack model $\vec{\theta}^\star_H$ is modified and is not necessarily equal to $\vec{\theta}'_H$. We assume that the attacker performs any of the following two types of data attacks:

1) **Data distortion:** We assume $\vec{\theta}^\star_H = \vec{\theta}'_H + \vec{z}$ for a random vector $\vec{z}$ with an arbitrary distribution with no positive probability mass in any proper linear subspace (e.g., multivariate Gaussian distribution).
2) **Data replay:** We assume $\vec{\theta}^\star_H = \vec{\theta}''_H$ such that $\vec{\theta}''$ satisfies $\mathbf{A}\vec{\theta}'' = \vec{p}''$ for an arbitrary power supply/demand vector $\vec{p}''$ such that $\vec{p}''_H = \vec{p}_H$. We assume that $\vec{p}''_{\bar{H}}$ is selected generally enough and is only known to the attacker. $\vec{p}''$ can be considered as the vector of supply/demand values from previous hours or days.

Notice that adversarial modification of the reported phase angles in H is not in the scope of this paper and is an interesting problem on its own. For example, see the recent work by Bienstock and Escobar [38].

Notation. Without loss of generality we assume that the indices are such that $V_H = \{1, 2, \ldots, |V_H|\}$ and $E_H = \{e_1, e_2, \ldots, e_{|E_H|}\}$. If X, Y are two subgraphs of G, $\mathbf{A}_{X|Y}$ and $\mathbf{A}_{V_X|V_Y}$ both denote the submatrix of the admittance matrix of G with rows from V_X and columns from V_Y. For instance, $\mathbf{A}$ can be written in any of the following forms,

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{H|H} & \mathbf{A}_{H|\bar{H}} \\ \mathbf{A}_{\bar{H}|H} & \mathbf{A}_{\bar{H}|\bar{H}} \end{bmatrix}, \mathbf{A} = \begin{bmatrix} \mathbf{A}_{G|H} & \mathbf{A}_{G|\bar{H}} \end{bmatrix}, \mathbf{A} = \begin{bmatrix} \mathbf{A}_{H|G} \\ \mathbf{A}_{\bar{H}|G} \end{bmatrix}.$$

3.3 Graph Theoretical Terms

In this paper, we use some graph theoretical terms most of which are borrowed from [39].

1. The matrix $\mathbf{A}$ can also be considered as the *weighted Laplacian matrix* of the graph.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

Subgraphs: Let X be a subset of the nodes of a graph G. $G[X]$ denotes the subgraph of G *induced* by X. We denote the complement of a set X by $\bar{X} = V \backslash X$.

The *neighbors*, *interior*, *boundary*, and *closure* of a subgraph S are defined and denoted by $N(S) := \{i \in V \backslash V_S | \exists j \in V_S : i \in N(j)\}$, $\text{int}(S) := \{i \in V_S | N(i) \subseteq V_S\}$, $\partial(S) := \{i \in V_S | N(i) \cap V_{\bar{S}} \neq \emptyset\}$, and $\text{cl}(S) := V_S \cup N(S)$, respectively.

Incidence Matrix: Assign arbitrary directions to the edges of G. The (node-edge) *incidence matrix* of G is denoted by $\mathbf{D} \in \{-1, 0, 1\}^{|V| \times |E|}$ and is defined as follows,

$$d_{ij} = \begin{cases} 0 & \text{if } e_j \text{ is not incident to node } i, \\ 1 & \text{if } e_j \text{ is coming out of node } i, \\ -1 & \text{if } e_j \text{ is going into node } i. \end{cases}$$

When we use the incidence matrix, we assume an arbitrary orientation for the edges unless we mention an specific orientation. $\mathbf{D}_H \in \{-1, 0, 1\}^{|V_H| \times |E_H|}$ is the submatrix of $\mathbf{D}$ with rows from V_H and columns from E_H.

4 Hardness

Using the notation provided in the previous section, the problem considered in this paper can be stated as follows: Given $\mathbf{A}, \vec{\theta}$, and $\vec{\theta}^\star$, detect the attacked area H and the set of line failures F. In this section, we study the computational complexity of this and related problems. To study the computational complexity of this problem, we consider a more general case of $\vec{\theta}^\star_H$ without any assumptions on the type of the data attack.

First, we prove that the problem of finding the set of line failures (F) solely based on the given the phase angles of the nodes before ($\vec{\theta}$) and after the attack ($\vec{\theta}'$) is NP-hard. We prove this by reduction from the *3-partition problem*.

Definition 1. Given a set $S = \{s_1, s_2, \ldots, s_{3k}\}$ of $3k$ elements and a bound B, such that $\sum_{i=1}^{3k} s_i = kB$ and for $1 \leq i \leq 3k$, $B/4 < s_i < B/2$, the *3-partition problem* is the problem of whether S can be partitioned into k disjoint sets $S_1, \ldots, S_k$ such that for $1 \leq i \leq k$, $\sum_{s_j \in S_i} s_j = B$ (note that each S_i must therefore contain exactly 3 elements from S).

Lemma 1 (Garey and Johnson [40]). The 3-partition problem is strongly NP-complete.

Lemma 2. Given $\mathbf{A}, \vec{\theta}$, and $\vec{\theta}'$, it is strongly NP-hard to determine if there exists a set of line failures F such that $\mathbf{A}'\vec{\theta}' = \mathbf{A}\vec{\theta}$.

Proof: We reduce the 3-partition problem to this problem. Assume S is a given set as described in Def. 1, we form a bipartite graph $G = (V, E)$ such that $V = X \cup Y$, $E = \{\{x, y\} | x \in X, y \in Y\}$, $X = \{1, \ldots, k\}$, and $Y = \{k+1, \ldots, 4k\}$. For all edges in G, we set the reactance values equal to 1. For each $i \in X$, we set $p_i = B$ and for each $j \in Y$ we set $p_j = -s_{j-k}$. Define the vector of phase angles $\vec{\theta}$ as follows:

$$\theta_i = \begin{cases} 0 & i \leq k \\ -s_{i-k}/k & i > k. \end{cases}$$

If $\mathbf{A}$ is the admittance matrix of G, it is easy to check that $\mathbf{A}\vec{\theta} = \vec{p}$. Now define $\vec{\theta}'$ as follows:

$$\theta'_i = \begin{cases} 0 & i \leq k \\ -s_{i-k} & i > k. \end{cases}$$

We prove that there exist a set of line failures F such that $\mathbf{A}'\vec{\theta}' = \vec{p}$ if, and only if, there exists a solution to the 3-partition problem.

First, lets assume that there exist a solution to the 3-partition problem such as $S_1, \ldots, S_k$. Set $E_S = \{\{i, j\} | s_{j-k} \in S_i\}$. We show that $F = E \backslash E_S$ implies $\mathbf{A}'\vec{\theta}' = \vec{p}$. Notice that $F = E \backslash E_S$ means that $G' = (V, E_S)$. Given the p_i and the reactance values, it is easy to check that the defined $\vec{\theta}'$ satisfies the DC power flow equations (1)-(2) in G'. Hence, $\mathbf{A}'\vec{\theta}' = \vec{p}$.

Now, lets assume there exist a set of line failures F such that $\mathbf{A}'\vec{\theta}' = \vec{p}$. Set $E_S = E \backslash F$ and $G' = (V, E_S)$. Given the phase angles $\vec{\theta}'$, it is easy to see that for any $\{i, j\} \in E_S$, $p_{ij} = s_{j-k}$. This implies that for $j \in Y$, at most one edge in E_S is incident to j. On the other hand, using (1), for any $i \in X$, $\sum_{j \in N(i)'} s_{j-k} = B$ in which by $N(i)'$ we mean the set of neighbors of node i in G'. Given that each node $j \in Y$ is incident to at most one edge in E_S, defining $S_i = \{s_{j-k} | j \in N(i)'\}$ for $1 \leq i \leq k$ gives a good solution to the 3-partition problem.

Hence, determining if there exist a set of line failures F is at least as hard as determining if the 3-partition problem has a solution, and therefore, it is an NP-hard problem in the strong sense. □

Corollary 1. Given $\mathbf{A}, \vec{\theta}$, and $\vec{\theta}'$, it is strongly NP-hard to find the set of line failures F, even if such a set exists.

In Corollary 1, we proved that given the phase angle of the nodes before and after the attack, it is NP-hard to detect the set of line failures F. In the following lemma, we show that even if the attack area H is known (since $\vec{\theta}'_H$ is not given) the problem remains NP-hard.

Lemma 3. Given $\mathbf{A}, \vec{\theta}, H$, and $\vec{\theta}'_{\bar{H}}$, it is strongly NP-hard to determine if there exist a set of line failures F in H and a vector $\vec{\theta}'_H$, such that $\mathbf{A}'\vec{\theta}' = \mathbf{A}\vec{\theta}$.

Proof: See Section 10. □

Corollary 2. Given $\mathbf{A}, \vec{\theta}, H$, and $\vec{\theta}'_{\bar{H}}$, it is strongly NP-hard to find the set of line failures F in H, even if such a set exists.

Finally, we prove that when the phase angles are modified ($\vec{\theta}^\star$) and therefore H is not known in advance, it is NP-hard to detect H and F. We assume that the attacked area cannot contain more than half of the nodes, otherwise this problem might have many solutions.

Lemma 4. Given $\mathbf{A}, \vec{\theta}$, and $\vec{\theta}^\star$, it is strongly NP-hard to determine if there exists a subgraph H_0 with $|V_{H_0}| \leq |V|/2$, a set of line failures F in H_0, and a vector $\vec{\theta}'_{H_0}$ such that $\mathbf{A}\vec{\theta} = \mathbf{A}' \begin{bmatrix} \vec{\theta}'_{H_0} \\ \vec{\theta}^\star_{\bar{H}_0} \end{bmatrix}$.

Proof: See Section 10. □

Corollary 3. Given $\mathbf{A}, \vec{\theta}$, and $\vec{\theta}^\star$, it is strongly NP-hard to find a subgraph H, a set of line failures F in H, and a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

vector $\vec{\theta}'_H$ such that $\mathbf{A}\vec{\theta} = \mathbf{A}' \begin{bmatrix} \vec{\theta}'_H \\ \vec{\theta}^\star_{\bar{H}} \end{bmatrix}$, even if such H, F exist.

Corollary 3 indicates that it is NP-hard to detect the line failures after an attack as described in Section 3 in general cases. However, in the next sections, we provide a polynomial-time algorithm to detect the attacked area H and the set of line failures F, and show based on simulations that it performs well in reasonable scenarios.

5 Attacked Area Approximation

In this section, we provide methods to approximate the attacked area after a cyber-physical attack as described in Subsection 3.2. By approximating the attacked area, we mean finding a subset of the nodes $V_S \subseteq V$ such that $V_H \subseteq V_S$ (i.e., V_S *contains the attacked area*). In the case of the perfect approximation, $V_S = V_H$. Once a set like V_S is found, we will detect the line failures is subgraph $S = G[V_S]$ in the next section.

In subsections 5.1 and 5.2, we first provide methods to contain the attacked area after the data distortion and replay attacks, respectively. We then combine these methods in the ATtacked Area Containment (ATAC) Module for containing the attacked area after both types of data attacks. Finally, given an area containing the attacked area, in subsection 5.4, we provide methods to improve the approximation of the attacked area.

5.1 Data Distortion

We first consider data distortion attacks. In particular, recall that we assume that $\vec{\theta}^\star_H = \vec{\theta}'_H + \vec{z}$ for a random vector $\vec{z}$ with an arbitrary distribution with no positive probability mass in any proper linear subspace. Since $\vec{\theta}^\star_H$ is the vector of the modified phase angles and there are also some line failures in H, it can be seen that $\mathbf{A}\vec{\theta}^\star \neq \vec{p}$. In Lemmas 5 and 6, we provide middle steps to prove that $\text{int}(\bar{H}) = V \backslash \text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})$ in Corollary 4. This demonstrates that nodes in $\text{int}(\bar{H})$ can be detected after a data distortion attack. For example, in Fig. 2, the green nodes that represent $\text{int}(\bar{H})$ can be detected by computing $V \backslash \text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})$.

Lemma 5. For any $i \in \text{int}(\bar{H})$, $\mathbf{A}_i\vec{\theta}^\star = p_i$.

Proof: Since $i \in \text{int}(\bar{H})$, therefore $a_{ij} = 0$ for all $j \in V_H$. Hence, $\mathbf{A}_i\vec{\theta}^\star = \mathbf{A}_{i|\bar{H}}\vec{\theta}^\star_{\bar{H}}$. On the other hand, since the attack is inside H, we know $\mathbf{A}_{i|\bar{H}} = \mathbf{A}'_{i|\bar{H}}$, and also $\vec{\theta}^\star_{\bar{H}} = \vec{\theta}'_{\bar{H}}$. Hence, $\mathbf{A}_i\vec{\theta}^\star = \mathbf{A}_{i|\bar{H}}\vec{\theta}^\star_{\bar{H}} = \mathbf{A}'_{i|\bar{H}}\vec{\theta}'_{\bar{H}} = \mathbf{A}'_i\vec{\theta}' = p_i$. □

Lemma 6. For any $i \in V \backslash \text{int}(\bar{H})$, $\mathbf{A}_i\vec{\theta}^\star \neq p_i$ almost surely.[2]

Proof: For any $i \in V \backslash \text{int}(\bar{H})$, there exists a node $j \in V_H$ such that $a_{ij} \neq 0$. Now since the set of solutions $\vec{x}$ to $\mathbf{A}_i\vec{x} = p_i$ is a measure zero set in $\mathbb{R}^n$ and $\theta^\star_j$ is a random modification of θ'_j, $\mathbf{A}_i\vec{\theta}^\star \neq p_i$ almost surely. □

Lemmas 5 and 6 indicate that given $\mathbf{A}, \vec{\theta}$, and $\vec{\theta}^\star$ one can find $\text{int}(\bar{H})$ by computing $V \backslash \text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})$.

Corollary 4. $\text{int}(\bar{H}) = V \backslash \text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})$, almost surely.

2. In probability theory, one says that an event happens almost surely, if it happens with probability one.

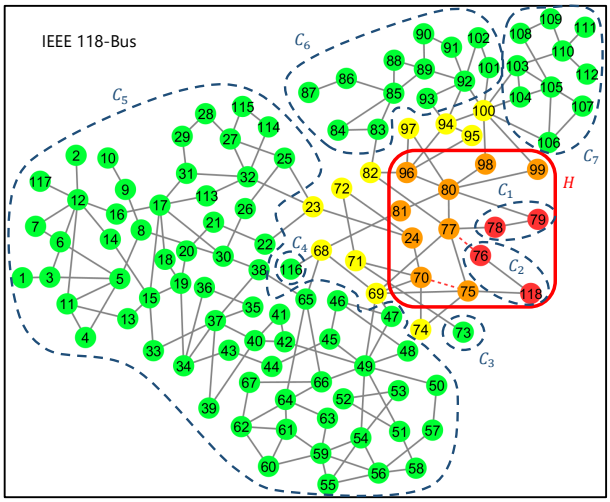


Fig. 2: H is an induced subgraph of G that represents the attacked area. Green, yellow, orange, and red nodes represent nodes in $\text{int}(\bar{H})$, $\partial(\bar{H})$, $\partial(H)$, and $\text{int}(H)$, respectively. $C_1, C_2, \ldots, C_7$ are the connected components of $G \backslash (\partial(\bar{H}) \cup \partial(H))$ that are used in Subsection 5.2 to detect the attacked area after a data replay attack.

Define $S_0 := G[\text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})]$. We know from Corollary 4 that $\text{int}(\bar{H}) = V_{\bar{S}_0}$ and from Lemma 6 that $V_H \subset V_{S_0}$. Therefore, S_0 clearly contains H. The following lemma demonstrates that $\text{int}(S_0)$ is a better approximation for V_H. For example, in Fig. 2, S_0 is represented by non-green nodes and $\text{int}(S_0)$ contains H plus nodes 95, 72, and 74. We use this lemma in Subsection 5.4 to improve the approximation of the attacked area.

Lemma 7. $V_H \subseteq \text{int}(S_0)$, almost surely.

Proof: Assume not. Then there exists a node $i \in V_H$ such that $N(i) \cap V_{\bar{S}_0} \neq \emptyset$. Assume $j \in N(i) \cap V_{\bar{S}_0} \neq \emptyset$, then with a similar argument as in the proof of Lemma 6, one can show that $\mathbf{A}_j\vec{\theta}^\star \neq p_j$ almost surely, which contradicts with $j \notin V_{S_0}$. Hence, $N(i) \cap V_{\bar{S}_0} = \emptyset$ and $V_H \subseteq \text{int}(S_0)$. □

5.2 Data Replay

In this subsection, we consider data replay attacks. Recall that we assume $\vec{\theta}^\star_H = \vec{\theta}''_H$ such that $\vec{\theta}''$ satisfies $\mathbf{A}\vec{\theta}'' = \vec{p}''$. The power supply/demand vector $\vec{p}''$ is arbitrarily selected such that $\vec{p}''_H = \vec{p}_H$, and $\vec{p}''_{\bar{H}}$ is selected generally enough.

The data replay attacks are harder to detect since the data seems to be correct locally. Again, one can easily see that $\mathbf{A}\vec{\theta}^\star \neq \vec{p}$, but here unlike the data distortion case, not all the nodes in H can be detected by checking $\mathbf{A}_i\vec{\theta}^\star \neq p_i$. The following lemmas and corollaries show why attacked area containment is more difficult after a data replay attack. In particular, Corollary 5 demonstrates that in the case of a data replay attack, $\text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})$ only reveals the nodes in the boundaries of the attacked area and its complement–i.e., $\partial(H)$ and $\partial(\bar{H})$. For example, in Fig. 2, yellow and orange nodes represent the boundary nodes in $\bar{H}$ and H, respectively.

Lemma 8. For any $i \in \text{int}(H) \cup \text{int}(\bar{H})$, $\mathbf{A}_i\vec{\theta}^\star = p_i$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

Proof: Similar to the proof of Lemma 5, it is easy to show that for any $i \in \text{int}(\bar{H})$, $\mathbf{A}_i\vec{\theta}^\star = p_i$. The only new part is to show the same for nodes in $\text{int}(H)$. So assume $i \in \text{int}(H)$, following the definition of the interior, it can be verified that $\mathbf{A}_i\vec{\theta}^\star = \mathbf{A}_{i|H}\vec{\theta}^\star_H$. On the other hand, since $\vec{\theta}^\star_H = \vec{\theta}''_H$ and $\vec{p}''_H = \vec{p}_H$, we can verify that $\mathbf{A}_{i|H}\vec{\theta}^\star_H = \mathbf{A}_{i|H}\vec{\theta}''_H = p''_i = p_i$. Hence, for all $i \in \text{int}(H)$ also $\mathbf{A}_i\vec{\theta}^\star = p_i$. □

Lemma 9. For any $i \in \partial(H) \cup \partial(\bar{H})$, $\mathbf{A}_i\vec{\theta}^\star \neq p_i$, almost surely.

Proof: The proof of this lemma is similar to the proof of Lemma 6. For any $i \in \partial(\bar{H})$, there exists a node $j \in V_H$ such that $a_{ij} \neq 0$. Now since the set of solutions $\vec{x}$ to $\mathbf{A}_i\vec{x} = p_i$ is a measure zero set in $\mathbb{R}^n$ and $\theta^\star_j = \theta''_j$ for a generally enough selected vector $p''_{\bar{H}}$, $\mathbf{A}_i\vec{\theta}^\star \neq p_i$ almost surely. A similar argument holds for $i \in \partial(H)$. □

Corollary 5. $\text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p}) = \partial(H) \cup \partial(\bar{H})$, almost surely.

From comparing Corollaries 4 and 5, one can see that in the replay attack case, $S_0 = G[\text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})]$ does not contain the attacked area H anymore. For example, in Fig. 2, S_0 contains only the yellow and orange nodes which does not contain the attacked area. In the following lemmas, we show how one can still contain the attacked area in this case.

Lemma 10. If $C_1, C_2, \ldots, C_k$ are the connected components of $G\backslash S_0$, then these connected components can be divided into two disjoint sets $\{i_1, i_2, \ldots, i_s\}$ and $\{j_1, j_2, \ldots, j_t\}$ such that $G[\text{int}(H)] = C_{i_1} \cup C_{i_2} \cup \cdots \cup C_{i_s}$ and $G[\text{int}(\bar{H})] = C_{j_1} \cup C_{j_2} \cup \cdots \cup C_{j_t}$.

Proof: It is a direct result of Corollary 5. □

Following Lemma 10, it can be seen that in Fig. 2, $G[\text{int}(H)] = C_1 \cup C_2$ and $G[\text{int}(\bar{H})] = C_3 \cup \cdots \cup C_7$.

Lemma 11. For two connected components C_i and C_j of $G\backslash S_0$, if $N(C_i) \cap N(C_j) \neq \emptyset$, then either $C_i \cup C_j \subseteq \text{int}(H)$ or $C_i \cup C_j \subseteq \text{int}(\bar{H})$.

Proof: From Lemma 10, for any i, either $C_i \subseteq G[\text{int}(\bar{H})]$ or $C_i \subseteq G[\text{int}(H)]$. If $C_i \subseteq G[\text{int}(\bar{H})]$ then $N(C_i) \subseteq \partial(\bar{H})$, and if $C_i \subseteq G[\text{int}(H)]$ then $N(C_i) \subseteq \partial(H)$. Hence, since $\partial(\bar{H}) \cap \partial(H) = \emptyset$, if $N(C_i) \cap N(C_j) \neq \emptyset$, then either $C_i \cup C_j \subseteq \text{int}(H)$ or $C_i \cup C_j \subseteq \text{int}(\bar{H})$. □

Following Lemma 11, one can see that connected components $C_1, C_2, \ldots, C_k$ can be combined into disjoint subgraphs $G_1, G_2, \ldots, G_t$ such that for any two of these subgraphs such as G_i and G_j, $N(G_i) \cap N(G_j) = \emptyset$. Moreover for any i, either $G_i \subseteq G[\text{int}(H)]$ or $G_i \subseteq G[\text{int}(\bar{H})]$.

For example, in Fig. 2, the connected components can be combined as $G_1 := C_1 \cup C_2$, $G_2 := C_4 \cup C_5$, $G_3 := C_3$, and $G_4 := C_6 \cup C_7$. It is easy to see that for any two of these subgraphs $N(G_i) \cap N(G_j) = \emptyset$. In the following lemma, we use this fact to contain the attacked area.

Lemma 12. There exists $1 \leq i \leq t$, such that $H \subset G\backslash G_i$. Moreover, $H \subseteq G[\text{int}(G\backslash G_i)]$.

Proof: The first part of the proof is the direct result of Lemmas 10 and 11. To prove the second part, notice that for any i, $S_0 \subset G\backslash G_i$. Therefore, for any i, $\partial(\bar{H}) \subset G\backslash G_i$. Hence, if $H \subset G\backslash G_i$, since $\partial(\bar{H}) \subset G\backslash G_i$, one can verify that $H \subseteq G[\text{int}(G\backslash G_i)]$. □

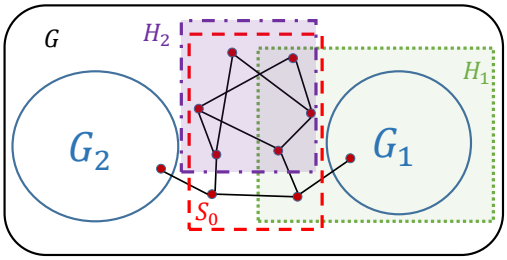


Fig. 3: An ambiguous scenario. Both a data replay attack on the attacked area H_1 or a data distortion attack on the attacked area H_2 result in the same $S_0 = G[\text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})]$.

Module 1: ATtacked Area Containment (ATAC)

Input: G, $\mathbf{A}$, $\vec{\theta}$, and $\vec{\theta}^\star$
1: Compute $\vec{p} = \mathbf{A}\vec{\theta}$
2: Compute $S_0 = G[\text{supp}(\mathbf{A}\vec{\theta}^\star - \vec{p})]$
3: Find the connected components $C_1, C_2, \ldots, C_k$ of $G\backslash S_0$
4: Using Lemma 11, combine the connected components with common neighbors to obtain $G_1, \ldots, G_t$ (sorted based on their size from largest to smallest)
5: Return $S_0, S_1 := G\backslash G_1, S_2 := G\backslash G_2, \ldots, S_t := G\backslash G_t$

Lemma 12 demonstrates that at least one of $G\backslash G_i$ contains the attacked area. Hence, one can use this fact to contain the attacked area after a data replay attack. For example, in Fig. 2, $G\backslash G_2$, $G\backslash G_3$, and $G\backslash G_4$, all contain H. Hence, any of them can be used to contain the attacked area. This clearly shows the difficulty in accurate detection of the attacked area after a replay attack compared to after a distortion attack. Since the system operator does not know the type of the attack in advance, in the next subsection, we combine the results of this subsection and the previous one to introduce a method for detecting the attacked area after both the data distortion and replay attacks.

5.3 The ATAC Module

Using the results in the previous subsections, here we introduce the ATtacked Area Containment (ATAC) Module for containing the attacked area after both types of data attacks. The main challenge is to distinguish between the two data attacks. As shown in Fig. 3, there are scenarios for which the data attack type cannot be recognized by simply looking at S_0. Hence, the ATAC Module does not return a single subgraph containing the attacked area but a series of possible subgraphs. In Sections 6 and 7, we show that by defining the *confidence of the solution*, an algorithm can go over all of these subgraphs until it detects the attacked area and the set of line failures with high confidence.

The steps of the ATAC Module are summarized in Module 1. As can be seen, S_0 is the first possible subgraph returned by the ATAC module, which is for the case when there is a data distortion attack. Then based on Lemma 12, $S_1 := G\backslash G_1, S_2 := G\backslash G_2, \ldots, S_t := G\backslash G_t$ are other possible areas containing the attacked area, if there is a replay attack. Notice that since $t < |V|$, therefore the ATAC module is a polynomial time algorithm.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

5.4 Improving Attacked Area Approximation

Assume that from the subgraphs returned by the ATAC Module, S^* is one of them that contains the attacked area H. Following Lemma 7 and Lemma 12, $S_a := G[\text{int}(S^*)]$ is a better approximation for the attacked area H. In order to find a more accurate approximation for H, we provide the following lemma which is similar to [3, Lemma 1].

Lemma 13. For a subgraph S, if $V_H \subseteq V_S$, then:

$$\mathbf{A}_{\bar{S}|G}(\vec{\theta} - \vec{\theta}') = 0. \tag{4}$$

Proof: Since all the line failures are inside H, and also $V_H \subseteq V_S$, therefore it can be seen that $\mathbf{A}_{\bar{S}|G} = \mathbf{A}'_{\bar{S}|G}$. On the other hand, $\mathbf{A}_{\bar{S}|G}\vec{\theta} = \vec{p}_{\bar{S}}$ and $\mathbf{A}'_{\bar{S}|G}\vec{\theta}' = \vec{p}_{\bar{S}}$. Hence, $\mathbf{A}_{\bar{S}|G}\vec{\theta} - \mathbf{A}'_{\bar{S}|G}\vec{\theta}' = 0$ and therefore $\mathbf{A}_{\bar{S}|G}(\vec{\theta} - \vec{\theta}') = 0$. □

Lemma 13 can be effectively used to estimate the phase angle of the nodes in S and to detect the attacked area H using these estimated values. The idea is to break (4) into parts that are known and unknown as follows:

$$\mathbf{A}_{\bar{S}|\bar{S}}(\vec{\theta}_{\bar{S}} - \vec{\theta}'_{\bar{S}}) + \mathbf{A}_{\bar{S}|S}(\vec{\theta}_S - \vec{\theta}'_S).$$

Notice that since $V_H \subseteq V_S$, therefore $\vec{\theta}'_{\bar{S}} = \vec{\theta}^{\star}_{\bar{S}}$. Hence, the only unknown variable in the equation above is $\vec{\theta}'_S$. Assume $\vec{y} \in \mathbb{R}^{|V_S|}$ is a solution to the following equation:

$$\mathbf{A}_{\bar{S}|S}\vec{y} = \mathbf{A}_{\bar{S}|\bar{S}}(\vec{\theta}_{\bar{S}} - \vec{\theta}^{\star}_{\bar{S}}) + \mathbf{A}_{\bar{S}|S}\vec{\theta}_S. \tag{5}$$

In the following lemma, we demonstrate that $\text{supp}(\vec{y} - \vec{\theta}^{\star}_S)$ can be used to estimate H.

Lemma 14. If $\vec{y}$ is a solution to (5), $V_H \subseteq \text{supp}(\vec{y} - \vec{\theta}^{\star}_S)$, almost surely.

Proof: Since $\vec{\theta}^{\star}_H$ is selected generally enough (for both the data distortion and replay attacks) for any $i \in H$, the only way $y_i = \theta^{\star}_i$ satisfying (5) is that $\mathbf{A}_{\bar{S}|i} = 0$. In that case any $y_i \in \mathbb{R}$ satisfies (5). So the set of solutions $\vec{y}$ such that $y_i = \theta^{\star}_i$ is a measure zero set and $y_i \neq \theta^{\star}_i$ almost surely. Hence, $V_H \subseteq \text{supp}(\vec{y} - \vec{\theta}^{\star}_S)$, almost surely. □

Following Lemma 14, if $\vec{y}$ is a solution to (5) for $S = S_a$, then $S_b := G[\text{supp}(\vec{y} - \vec{\theta}^{\star}_{S_a})]$ is a better approximation for the attacked area H. Fig. 4 shows the difference between S^*, S_a, and S_b in approximating the attacked area for the case of a distortion attack and if $S^* = S_0$ (recall that S_0 is the first set returned by the ATAC Module). It can be seen that S_b is not exactly equal to H even in the case of the data distortion attack.

The following lemma demonstrates when S_b is exactly equal to H.

Lemma 15. For a subgraph S such that $V_H \subseteq V_S$, if $V_S \backslash V_H \subseteq \partial(S)$ and there is a matching between the nodes in $\bar{S}$ and $\partial(S)$ that covers all the nodes in $\partial(S)$, then $G[\text{supp}(\vec{y} - \vec{\theta}^{\star}_S)] = H$, in which $\vec{y}$ is the solution to (5).

Proof: If there is a matching between the nodes inside and outside of H that covers all the nodes in $\partial(S)$, one can prove that $\mathbf{A}_{\bar{S}|\partial(S)}$ has linearly independent columns, almost surely (see [3, Corollary 2]). Moreover, it is easy to see that $\mathbf{A}_{\bar{S}|\text{int}(S)} = 0$. Hence, if $\vec{y}$ is a solution to (5), $y_{\partial(H)} = \vec{\theta}'_{\partial(H)}$. Now since $V_S \backslash V_H \subseteq \partial(S)$, for any i in $V_S \backslash V_H$, $y_i = \theta'_i = \theta^{\star}_i$. On the other hand, since $\vec{\theta}^{\star}_H$ are selected generally enough, one can verify that for any $i \in H$, $y_i \neq \theta^{\star}_i$, almost surely. Therefore, $G[\text{supp}(\vec{y} - \vec{\theta}^{\star}_S)] = H$, almost surely. □

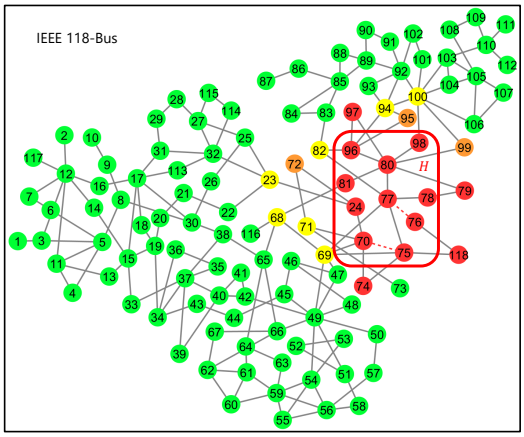


Fig. 4: H is an induced subgraph of G that represents the attacked area. If the data attack type is *data distortion* and $S^* = S_0$, then the red, orange, yellow, and green nodes represent the nodes in S_b, $S_a \backslash S_b$, $S^* \backslash S_a$, $G \backslash S^*$ as defined in Subsection 5.4, respectively.

6 Line Failures Detection

In the previous section, we provided methods to find a good approximation S for the attacked area H. In this section, we provide a method to detect line failures inside S. For this reason, we use and build on the idea introduced in [3]. It was proved in [3] that if the attacked area H is known, then there always exists feasible vectors $\vec{x} \in \mathbb{R}^{|E_H|}$ and $\vec{y} \in \mathbb{R}^{|V_H|}$ satisfying the conditions of the following optimization problem such that $\text{supp}(\vec{x}) = F$ and $\vec{y} = \vec{\theta}'_H$:

$$\begin{aligned} &\min_{\vec{x},\vec{y}} \|\vec{x}\|_1 \text{ s.t.} \\ &\mathbf{A}_{H|H}(\vec{\theta}_H - \vec{y}) + \mathbf{A}_{H|\bar{H}}(\vec{\theta}_{\bar{H}} - \vec{\theta}'_{\bar{H}}) = \mathbf{D}_H\vec{x} \\ &\mathbf{A}_{\bar{H}|H}(\vec{\theta}_H - \vec{y}) + \mathbf{A}_{\bar{H}|\bar{H}}(\vec{\theta}_{\bar{H}} - \vec{\theta}'_{\bar{H}}) = 0. \end{aligned} \tag{6}$$

Notice that the optimization problem (6) can be solved efficiently using Linear Program (LP). It is proved in [3] that under some conditions on H and the set of line failures F, the solution to (6) is unique, therefore the relaxation is exact and the set of line failures can be detected by solving (6). In particular, it is proved in [3] that if H is acyclic and there is a matching between the nodes in H and $\bar{H}$ that covers H, the solution to (6) is unique for any set of line failures.

For example, in Fig. 4, it is easy to see that H is acyclic. Moreover, it can be verified that $M = \{\{96, 97\}, \{80, 99\}, \{98, 100\}, \{78, 79\}, \{76, 118\}, \{77, 82\}, \{75, 74\}, \{70, 69\}, \{24, 72\}, \{81, 68\}\}$ is a matching between the nodes in H and $\bar{H}$ that covers H. Hence, if H is known–i.e., we could detect the attacked area accurately–solving (6) recovers the phase angles and detect the line failures.

Since the conditions on H and F as described in [3] may not always hold for the exactness of the line failures detection using (6), it cannot be used in general cases to detect line failures. To address this issue, here, we introduce a randomized version of (6).

Assume that $\mathbf{W} \in \mathbb{R}^{|E_S| \times |E_S|}$ is a diagonal matrix. We show that the solution to the following optimization

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

Module 2: LIne Failures Detection (LIFD)

Input: G, $\mathbf{A}$, $\vec{\theta}$, S, T, and $\vec{\theta}^\star$
1: Compute $\vec{p} = \mathbf{A}\vec{\theta}$
2: Compute a solution $\vec{x}, \vec{y}$ to (7) for $\mathbf{W} = \mathbf{I}$
3: Set $F^\dagger = \text{supp}(\vec{x})$ and $\vec{\theta}^\dagger_S = \vec{y}$
4: **while** $c(F^\dagger, \vec{\theta}^\dagger_S) < 99.99\%$ & counter$< T$ **do**
5: counter++
6: Draw random numbers $w_1, w_2, \ldots, w_{|V_S|}$ from an exponential distribution with rate $\lambda = 1$
7: Compute a solution $\vec{x}, \vec{y}$ to (7) for $\mathbf{W} = \text{diag}(w_1, w_2, \ldots, w_{|V_S|})$
8: Set $F^\dagger = \text{supp}(\vec{x})$ and $\vec{\theta}^\dagger_S = \vec{y}$
9: **if** $c(F^\dagger, \vec{\theta}^\dagger_S) > 99.99\%$ **then**
10: **return** $F^\dagger, \vec{\theta}^\dagger_S$
11: **else**
12: **return** $F^\dagger, \vec{\theta}^\dagger_S$ with maximum $c(F^\dagger, \vec{\theta}^\dagger_S)$ in all iterations

problem can detect line failures in S accurately for a "good" matrix $\mathbf{W}$:

$$\begin{aligned} &\min_{\vec{x},\vec{y}} \|\mathbf{W}\vec{x}\|_1 \text{ s.t.} \\ &\mathbf{A}_{S|S}(\vec{\theta}_S - \vec{y}) + \mathbf{A}_{S|\bar{S}}(\vec{\theta}_{\bar{S}} - \vec{\theta}'_{\bar{S}}) = \mathbf{D}_S\vec{x} \qquad (7) \\ &\mathbf{A}_{\bar{S}|S}(\vec{\theta}_S - \vec{y}) + \mathbf{A}_{\bar{S}|\bar{S}}(\vec{\theta}_{\bar{S}} - \vec{\theta}'_{\bar{S}}) = 0. \end{aligned}$$

The idea behind optimizing the weighted norm-1 of vector $\vec{x}$ is to be able to detect the line failures when the solution to (6) does not detect the correct set of line failures but a small disturbance results in the correct detection.

Before we demonstrate the effectiveness of the optimization (7) in detecting line failures, we provide a metric for measuring the *confidence of a solution*. In a subgraph S, assume $F^\dagger = \text{supp}(\vec{x})$ and $\vec{\theta}^\dagger_S = \vec{y}$ are the set of detected line failures and the recovered phase angles using the solution to (7). Also assume that $\mathbf{A}^\dagger$ is the admittance matrix after removing the lines in $F^\dagger$ and define $\vec{p}^\dagger := \mathbf{A}_{G|\bar{S}}\vec{\theta}'_{\bar{S}} + \mathbf{A}^\dagger_{G|S}\vec{\theta}^\dagger_S$. Notice that $\vec{x}$ and $\vec{y}$ satisfying (7) does not necessarily imply $\vec{p}^\dagger = \vec{p}$. Hence, one can use this difference to compute the confidence of a solution as follows.

Definition 2. The *confidence of the solution* is denoted by $c(F^\dagger, \vec{\theta}^\dagger_S)$ and defined as:

$$c(F^\dagger, \vec{\theta}^\dagger_S) := (1 - \|\vec{p}^\dagger - \vec{p}\|_2/\|\vec{p}\|_2)^+ \times 100, \qquad (8)$$

in which $(z)^+ := \max(0, z)$.

The confidence of the solution, simply shows how much the solution is consistent with the part of the observed data that we detected as correct. In other words, it checks if the solution fulfills the conditions provided in Lemma 4.

The confidence of the solution along with a random selection of the weight matrix $\mathbf{W}$ in (7) can be used to detect line failures that cannot be detected using (6). The idea is to repeatedly solve (7) using a random weight matrix until the confidence of the solution for $F^\dagger = \text{supp}(\vec{x})$ and $\vec{\theta}^\dagger_S = \vec{y}$ is 100% or reach a maximum number of iterations (T). Here, we consider the case when the diagonal entries of matrix $\mathbf{W}$ are randomly selected from an exponential distribution. This approach is summarized in Module 2 as the LIne Failures Detection (LIFD) Module.

Through the rest of this section, we demonstrate why the LIFD Module is effective and when the number of iterations (T) is enough to be polynomial in terms of the input size to make sure that it finds the line failures accurately.

Lemma 16. Assume $w_1, w_2, \ldots, w_m$ are i.i.d. exponential random variables, then for $1 \le k \le m-1$:

$$Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) = \frac{\sum_{j=k}^{m-1} \binom{m-1}{j}}{2^{m-1}}.$$

Proof: See Section 10. □

Corollary 6. Assume $w_1, w_2, \ldots, w_m$ are i.i.d. exponential random variables, then for $k \le m/2 + \Theta(\sqrt{m})$:

$$Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) = \Omega(\frac{1}{\sqrt{m}}).$$

Proof: See Section 10. □

Lemma 17. If $S = H$, H is a cycle with m nodes and edges, and there is a matching between the nodes inside and outside of H that covers all the inside nodes, then any set of line failures of size k can be found by the LIFD Module for expectedly $T = 2^{m-1}/(\sum_{j=k}^{m-1} \binom{m-1}{j})$. Moreover, if $k \le m/2 + \Theta(\sqrt{m})$, then LIFD Module can detect line failures for $T = O(\sqrt{m})$.

Proof: First, one can see that if $S = H$, and there is a matching between the nodes inside and outside of H that covers all the inside nodes, then $\mathbf{A}_{\bar{S}|S} = \mathbf{A}_{\bar{H}|H}$ has uniquely independent columns, almost surely [3, Corollary 2]. Hence, the solution $\vec{y}$ to (7) is unique and $\vec{y} = \vec{\theta}'_H$. Therefore, we can assume that $\vec{\theta}'$ is given. Without loss of generality assume that $F = \{e_1, \ldots, e_k\}$. We prove that the solution $\vec{x}$ to (7) is unique and $\text{supp}(\vec{x}) = F$, if $\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i$, in which $\mathbf{W} = \text{diag}(w_1, \ldots, w_m)$.
Without loss of generality, assume that $\mathbf{D}_H$ is the incidence matrix of H when lines of H are oriented clockwise. Since H is connected, it is known that $\text{rank}(\mathbf{D}_H) = m - 1$ [41, Theorem 2.2]. Therefore, $\dim(\text{Null}(\mathbf{D}_H)) = 1$. Suppose $\vec{z} \in \mathbb{R}^{|E_H|}$ is the all one vector. It can be verified that $\mathbf{D}_H\vec{z} = 0$. Since $\dim(\text{Null}(\mathbf{D}_H)) = 1$, $\vec{z}$ forms a basis for the null space of $\mathbf{D}$. Now suppose $\vec{x}^\dagger$ is a solution to $\mathbf{A}_{H|G}(\vec{\theta} - \vec{\theta}') = \mathbf{D}_H\vec{x}$ such that $\text{supp}(\vec{x}^\dagger) = F$ (from [3, Lemma 2], we know that such a solution exists). Since $\vec{z}$ forms a basis for $\text{Null}(D)$, all other solutions of $\mathbf{A}_{H|G}(\vec{\theta} - \vec{\theta}') = \mathbf{D}_H\vec{x}$ can be written in the form of $\vec{x}^\dagger + c\vec{z}$. We want to prove that if $\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i$, then for any $c \in \mathbb{R}\backslash\{0\}$, $\|\mathbf{W}\vec{x}^\dagger\|_1 < \|\mathbf{W}(\vec{x}^\dagger + c\vec{z})\|_1$. Since $\text{supp}(\vec{x}^\dagger) = F$, $x^\dagger_1, x^\dagger_2, \ldots, x^\dagger_k$ are the only nonzero elements of $\vec{x}^\dagger$. Moreover $W_d := \sum_{i=k+1}^{m} w_i - \sum_{i=1}^{k} w_i > 0$. Hence,

$$\begin{aligned} \|\mathbf{W}(\vec{x}^\dagger + c\vec{z})\|_1 &= \sum_{i=1}^{k} w_i|x^\dagger_i - c| + |c| \sum_{i=k+1}^{m} w_i \\ &= \sum_{i=1}^{k} w_i(|x^\dagger_i - c| + |c|) + |c|W_d \\ &\ge \sum_{i=1}^{k} w_i|x^\dagger_i| + |c|W_d > \sum_{i=1}^{k} w_i|x^\dagger_i| = \|\mathbf{W}\vec{x}^\dagger\|_1. \end{aligned}$$

Therefore, the solution $\vec{x}$ to (7) is unique and $\text{supp}(\vec{x}) = F$, if $\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i$. One the other hand, from Lemma 16, $Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) = \frac{\sum_{j=k}^{m-1} \binom{m-1}{j}}{2^{m-1}}$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

Algorithm 1: REcurrent Attack Containment and deTection (REACT)

Input: G, $\mathbf{A}$, $\vec{\theta}$, $\vec{\theta}^{\star}$, and T
Compute $\vec{p} = \mathbf{A}\vec{\theta}$
Obtain $S_0, S_1, \ldots, S_t$ using the ATAC Module
for $i = 1$ to t **do**
  Compute $S_a = G[\text{int}(S_i)]$
  if (5) is feasible for $S = S_a$ **then**
    Find a solution $\vec{y}$ to (5) for $S = S_a$
  else
    continue
  Compute $S_b = G[\text{supp}(\vec{y} - \vec{\theta}^{\star}_S)]$
  Set $S = S_b$ as an approximation for the attacked area H
  Compute a solution $\vec{x}, \vec{y}$ to (7) for $\mathbf{W} = \mathbf{I}$
  Set $F^{\dagger} = \text{supp}(\vec{x})$ and $\vec{\theta}^{\dagger}_S = \vec{y}$
  if $c(F^{\dagger}, \vec{\theta}^{\dagger}_S) < 99.99\%$ **then**
    Obtain $F^{\dagger}, \vec{\theta}^{\dagger}_S$ from module LIFD for inputs S and T
  if $c(F^{\dagger}, \vec{\theta}^{\dagger}_S) > 99.99\%$ **then**
    return $H = \text{supp}(\vec{\theta}^{\dagger}_S - \vec{\theta}^{\star}_S)$ as the detected attacked area and $F^{\dagger}, \vec{\theta}^{\dagger}_H$ as the detected line failures and recovered phase angle of the nodes inside H
return S and $F^{\dagger}, \vec{\theta}^{\dagger}_S$ with maximum $c(F^{\dagger}, \vec{\theta}^{\dagger}_S)$ in all iterations

Hence, expectedly $\frac{2^{m-1}}{\sum_{j=k}^{m-1}\binom{m-1}{j}}$ number of iterations (T) should be enough to satisfy this inequality. Corollary 6 also gives the expected number of iterations needed when $k \leq m/2 + \Theta(\sqrt{m})$. □

Lemma 17 clearly demonstrates the effectiveness of using a weight matrix **W** in (7). It was previously proved in [3] that if H is a cycle and there is a matching between the nodes inside and outside of H that covers all the inside nodes, then for any set of line failures of size *less than half of the lines in* H, $\text{supp}(\vec{x})$ of the solution $\vec{x}$ to (6) exactly reveals the set of line failures. However, for the line failures with the size more than half of the lines in H, this approach comes short. In these cases, Lemma 17 indicates that solving (7) for random matrices **W** for polynomial number of times can lead to the correct detection.

Although providing a similar analytical bound for T to ensure detecting line failures in general cases is very difficult, in Section 8, we numerically show that small values of T is enough to detect line failures in more complex attacked areas as well.

7 REACT Algorithm

In this section, we present the REcurrent Attack Containment and deTection (REACT) Algorithm based on the results presented in the previous sections. The steps of the REACT Algorithm are summarized in Algorithm 1.

The REACT Algorithm first obtains a set of possible subgraphs $S_0, S_1, \ldots, S_t$ that may contain the attacked area H using the ATAC Module. Then, for each subgraph S_i using the results in Subsection 5.4, it improves the approximation of the attacked area. In particular, it first computes $S_a = G[\text{int}(S_i)]$ and then finds a solution to (5) for $S = S_a$. If (5) is not feasible, then it means that S_i does not contain the attacked area H, and therefore, the algorithm goes to the next iteration and tries the next possible subgraph. If (5) has a feasible solution $\vec{y}$, it obtains a better approximation of the attacked area H by computing $S_b = G[\text{supp}(\vec{y} - \vec{\theta}^{\star}_S)]$ (Lemma 14).

Then, it solves the optimization (7) for $\mathbf{W} = \mathbf{I}$, in which **I** is the identity matrix. Notice that this is basically similar to solving (6). Then it checks the confidence of the solution $c(F^{\dagger}, \vec{\theta}^{\dagger}_S)$. If it is less than 99.99%, it calls the LIFD Module to obtain another solution $F^{\dagger}, \vec{\theta}^{\dagger}_S$. Finally, it checks whether the confidence of the solution is $c(F^{\dagger}, \vec{\theta}^{\dagger}_S) > 99.99\%$. If so, it approximates the attacked area H using this solution and returns $F^{\dagger}, \vec{\theta}^{\dagger}_H$.

If the REACT Algorithm cannot find a solution with confidence greater than 99.99%, it returns a solution with the highest confidence between all the solutions obtained in all the iterations.

Notice that the REACT Algorithm is a polynomial time algorithm. Therefore, it cannot return the correct solution to an NP-hard problem in all cases. However, in the next section we numerically demonstrate that it performs very well in reasonable settings.

8 Numerical Results

In this section, we evaluate the performance of the REACT Algorithm in detecting the attacked area and recovering the information after a cyber-physical attack as described in Section 3.2. We consider two attacked areas H_1 and H_2 within the IEEE 300-bus system [4] as depicted in Fig. 5. H_1 has 15 nodes and 16 edges, and H_2 which contains H_1, has 31 nodes and 41 edges. It can be verified that none of these two subgraphs are acyclic and there is no matching between the nodes inside and outside of these two subgraphs that covers their insides nodes. *Hence, the methods provided in [3] cannot recover the information inside these areas even when the attacked areas are known in advance.*

For the physical part of the attack, we consider all single line failures, and 100 samples of all double and triple line failures within H_1 and H_2. Figs. 7 and 8 illustrate the REACT Algorithm's performance after these attacks. In the Algorithm, we set $T = 20$ so that the while loop in the LIFD Module runs only for 20 iterations.

Fig. 7 shows the performance of the REACT Algorithm in detecting the attacked area and recovering the information after data distortion and data replay attacks on the attacked area H_1 accompanied by single, double, and triple line failures. As can be seen in Fig. 7(a), the REACT Algorithm can exactly detect the attacked area after all attack scenarios under both the distortion attack and the replay attack. Hence, the performance of the REACT Algorithm is almost the same in detecting line failures and recovering the phase angles after both data attack scenarios.

Fig. 7(b) shows the average number of False Negatives (FN) and False Positives (FP) in detecting line failures. As can be seen, the REACT Algorithm can detect line failures with very low average number of FNs and FPs. Moreover, as it is shown in Fig. 7(c), the REACT Algorithm exactly detects single, double, and triple line failures in 94%, 87%, and 82% of the cases, respectively.

Fig. 7(d) shows the average running time of the REACT Algorithm in detecting all attacked scenarios in this case. Our system has an Intel Core i7-2600 3.40GHz CPU and 16GB RAM. One can see that the running time of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

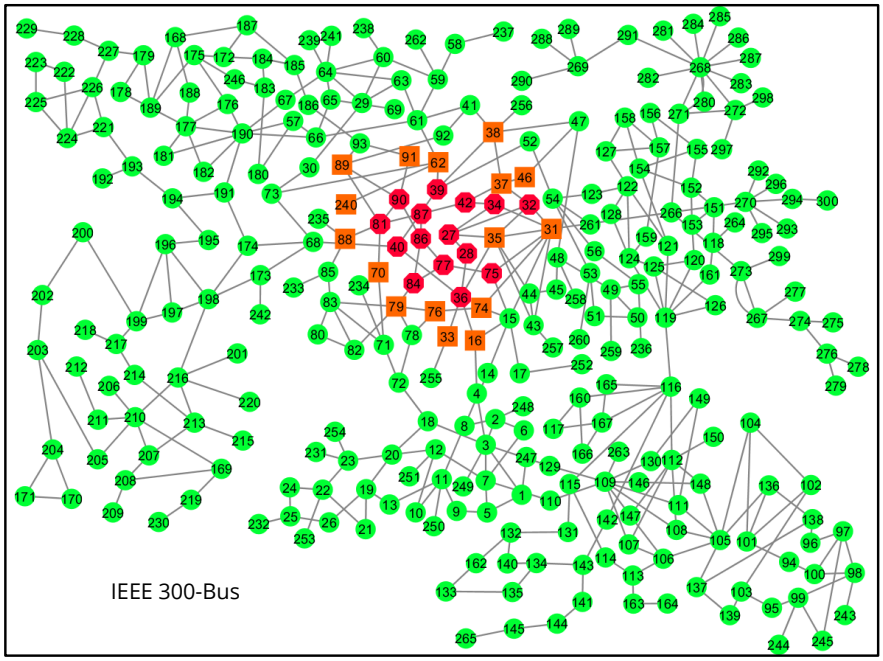
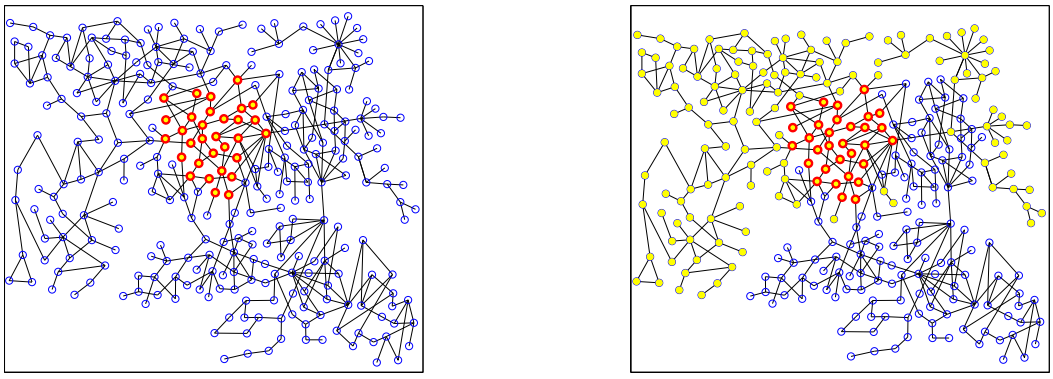


Fig. 5: The two attacked areas in the IEEE 300-bus systems that are used in simulations. The red octagon nodes are the nodes in H_1 and H_2, and the orange square nodes are the nodes that are only in H_2.



(a) Data Distortion Attack

(b) Data Replay Attack

Fig. 6: The difference in difficulty of detecting the attacked area after a data distortion attack and a data replay attack on the attacked area H_2 accompanied by a triple line failure within H_2. The yellow filled nodes represent the nodes in the detected attacked area by the REACT Algorithm, the nodes with a thick red border represent the nodes in H_2 that are actually attacked, and blue empty nodes represent the rest of the nodes.

REACT Algorithm is very low. The average confidence of the solutions are also shown in Fig. 7(e). As can be seen, despite few false negatives and positives in detecting line failures, the solutions obtained by the REACT Algorithm have very high confidence which means that the REACT Algorithm barely missed finding the correct solution.

Finally, Fig. 7(f) shows the average percentage error in the recovered phase angles. It can be seen that the phase angles inside the attacked area can be recovered with less than 3%, 5%, and 7% error after the single, double, and triple line failures, respectively.

As we observed in Fig. 7, when the attacked area is relatively small, the REACT Algorithm performs very similarly after the two types of data attack. However, as it can be clearly seen in Fig. 8, it is not the case as the attacked area becomes larger. Before we analyze the results provided in Fig. 8, in order to better show the difficulty of detecting the attacked area after a data replay attack, we depicted in Fig. 6 one of the analyzed attacked scenarios in Fig. 8. As can be seen in Fig. 6(a), the REACT Algorithm can exactly detect the attacked area after a data distortion attack on H_2 which is accompanied by a triple line failure. However, it may have difficulties detecting the attacked area after a data replay attack on the same area with the same set of line failures. Recall from Subsection 5.2 that the main reason for this is the difficulty of distinguishing between the nodes in $\text{int}(H)$ and $\text{int}(\bar{H})$.

Fig. 8(a) shows the extra nodes that are incorrectly detected by the REACT Algorithm as part of the attacked area. As can be seen, in the case of the data distortion attack, the number of line failures do not significantly affect the performance of the REACT Algorithm. However, in the case

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

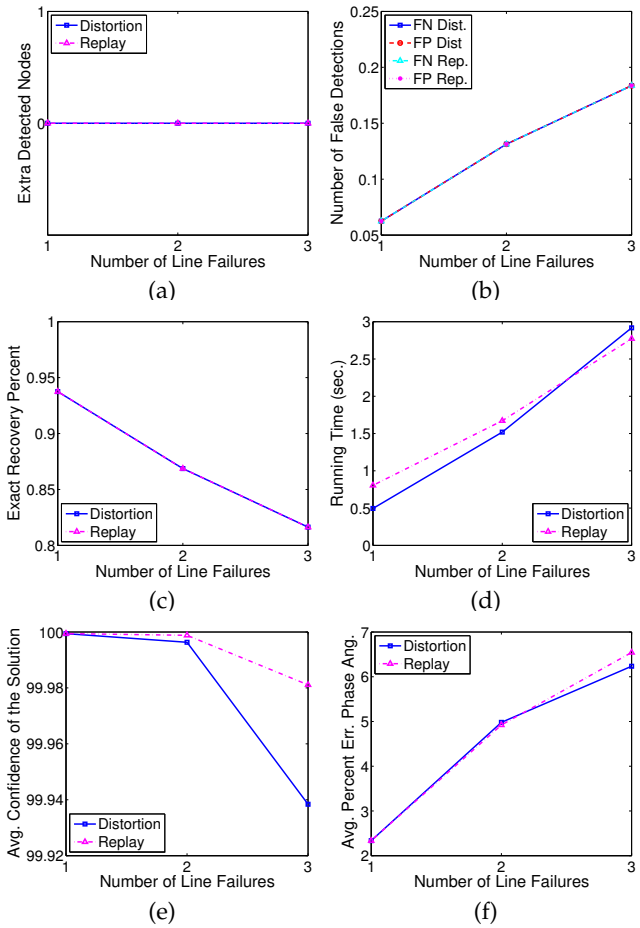


Fig. 7: The REACT Algorithm's performance in detecting the attacked area and recovering the information after data distortion and replay attacks on the attacked area H_1 accompanied by single, double, and triple line failures. (a) Average number of extra nodes detected as attacked in detecting the attacked area, (b) average number of false positives and negatives in detecting line failures, (c) percentage of the cases with exact line failures detection, (d) running time of the algorithm, (e) average confidence of the solutions, and (f) average error in recovered phase angles.

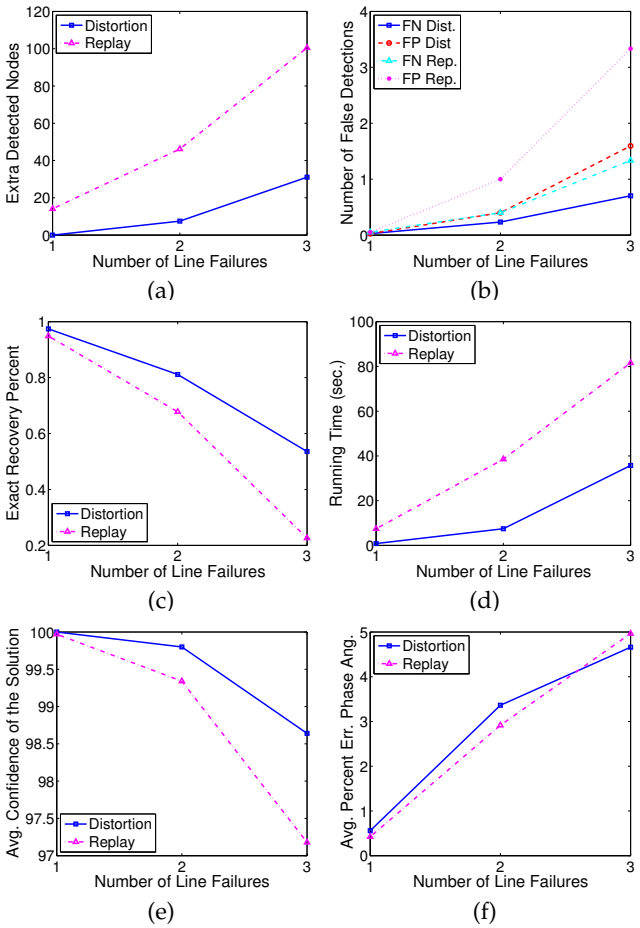


Fig. 8: The REACT Algorithm's performance in detecting the attacked area and recovering the information after data distortion and replay attacks on the attacked area H_2 accompanied by single, double, and triple line failures. (a) Average number of extra nodes detected as attacked in detecting the attacked area, (b) average number of false positives and negatives in detecting line failures, (c) percentage of the cases with exact line failures detection, (d) running time of the algorithm, (e) average confidence of the solutions, and (f) average error in recovered phase angles.

of the replay attack, as the number of line failures within the attacked area increases, the REACT Algorithm provides less accurate approximation of the attacked area.

Despite its difficulty in detecting the attacked area after a data replay attack, Figs. 8(b) and 8(c) demonstrate that the REACT Algorithm detects the line failures relatively accurately. For example, the REACT Algorithm accurately detects the single and double line failures in 95% and 65% of the cases, respectively. This clearly demonstrates the advantage of the optimization (7) that is used in the LIFD module compared to (6) that suggested in [3]. Since (6) fails to detect the line failures accurately as the attacked zone contains more cycles and more internal nodes (i.e., nodes that are not connected to any nodes outside of the attacked zone), as it is the case here.

As can be seen in Fig. 8(d), the running time of the REACT Algorithm increases as the size of the attacked area increases. However, it still detects line failures much faster than existing brute force methods [29], [30], [32], [42], [43] which their running time increases exponentially as the number of line failures and the total number of possibilities increase. Notice that [29], [30], [32], [42], [43] do not deal with the case that the attack area is unknown. The comparison is only between the running time of the LIFD module and the brute force methods. The exponential running times of the brute force algorithms become more problematic in the data replay attack case. Since in this case, as described in Section 5, the attacked zone cannot be approximated independently of the line failures detection module and this module should be called for different possible attacked zones in order to detect the one that contains the actual attacked zone. Hence, the exponential running time of the brute force search algorithms make them completely impractical for this case.

Similar to the previous attack scenario, one can see in Fig. 8(e) that the confidence of the solutions obtained by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

the REACT Algorithm are very high. It means that in these attack scenarios, many good solutions exist near the optimal solution. This demonstrates another difficulty of dealing with recovery of information after a cyber-physical attack on the power grid.

Finally, Fig. 8(f) indicates that the REACT Algorithm performs very well in recovering the phase angles in this case as well. As can be seen, for both the data distortion and the data replay attacks accompanied by single, double, and triple line failures, the REACT Algorithm recovers the phase angles with less than 5% error.

Overall, the simulation results in this section demonstrate that the REACT Algorithm performs very well in detecting the attacked area and the line failures when the attacked area is relatively small. As the attacked area becomes larger, the Algorithm still performs very well in detecting the attacked area after a distortion data attack. However, it may face difficulties providing an accurate approximation of the attacked area after a replay attack. Despite this, in both data attack scenarios, it detects line failures relatively well. One of the important observations in this section is that the LIFD module outperforms the methods provided in [3] for detecting line failures with an slight increase in the running time, since it needs to find a solution to (7) several times instead of once. The results in this section clearly demonstrate that in the attacked areas H_1 and H_2 that do not have the conditions provided [3], the LIFD Module can still detect the line failures relatively accurately with less than 20 iterations. In most of theses cases, the LIFD Module detects the line failures within much fewer number of iterations.

9 Conclusion

In this paper, we considered a model for cyber-physical attacks on power grids focusing on both data distortion and data reply attacks. We proved that the problem of detecting the line failures after such an attack is NP-hard in general and even when the attacked area is known. However, using the algebraic properties of the DC power flows, we developed the polynomial time REACT Algorithm for approximating the attacked area and detecting the line failures after a cyber-physical attack on the grid. We numerically showed that the REACT Algorithm obtains accurate results when there are few number of line failures and the attacked area is small. We showed that as the attacked area becomes larger and the number of line failures increases, the REACT Algorithm faces some difficulties but still can approximate the attacked area and detect line failures with few false negatives and positives.

The goal of this paper was to provide a theoretical foundation for the problem of attacked area and line failures detection after a cyber-physical attack on the power grid. Hence, in this work, we neglected the measurement noise in our analysis and also considered the availability of PMUs at all the nodes. Nevertheless, we demonstrated that this problem is already very challenging without considering these constraints. Extending the results and methods of this paper to the cases where the measurements are noisy and there are limited number of PMUs in the system is part of our future work.

Although the DC power flows only provide an approximation for the more accurate AC power flows, since the ATAC Module for detecting the attacked area mostly depends on the flow conservation checks at each node, the ATAC Module can be easily applied under the AC power flows as well. Moreover, the weight randomization technique and the confidence metric used in the LIFD Module can also be extended to the AC power flows using the methods provided in a recent paper [34]. Extending the results provided in this paper to the transient state of power grids, however, is of particular interest to the power systems community and is part of our future work.

As we proved in Section 6, when the attacked area is a cycle, the weight randomization technique in the LIFD Module can detect the line failures accurately in the expected polynomial running time. Extending this analytical result to the attacked areas with arbitrary topology is an interesting and challenging future work.

Finally, we analytically and numerically showed that the data replay attacks are harder to deal with than the data distortion attacks. Moreover, It is possible for an adversary to devise more sophisticated attacks to further obscure the system's state. We believe that by trading running time for accuracy, we may be able to improve the accuracy of the REACT Algorithm in detecting the attacked area and the line failures after replay attacks. However, depending on the situation, a faster but approximately accurate algorithm may be more desirable than a more accurate but slower one. Careful speculation of such trade-offs and exploring more sophisticated attacks are part of our future work.

10 Omitted Proofs

Proof of Corollary 1: It is easy to see that if one can find a set of line failures F with an algorithm, the output of that algorithm can be used here to verify the correctness and existence of such a set as well. Therefore, this problem is at least as hard as the existence problem. □

Proof of Lemma 3: The idea of the proof is very similar to the proof of Lemma 2. Again we reduce the 3-partition problem with a given set S as described in Def. 1 to this problem. Consider sets $X_1 = \{1, \dots, k\}$, $X_2 = \{k+1, \dots, 2k\}$, $Y_2 = \{2k+1, \dots, 5k\}$, $Y_1 = \{5k+1, \dots, 8k\}$. We form a bipartite graph $G = (V, E)$ such that $V = X_1 \cup X_2 \cup Y_2 \cup Y_1$ and $E = \{\{i, k+i\} | 1 \leq i \leq k\} \cup \{\{x, y\} | x \in X_2, y \in Y_2\} \cup \{\{j, j+3k\} | 2k+1 \leq j \leq 5k\}$. Notice that the defined bipartite graph here is very similar to the one defined in the proof of Lemma 2 except that here for each node in X_2 and Y_2 there exist a dummy node in X_1 and Y_1, accordingly, that is directly connected to its counterpart. We set $H = G[X_2 \cup Y_2]$. It is easy to see that H has exactly the same topology as the graph G in the proof of Lemma 2. Again for all edges in G, we set the reactance values equal to 1. For each $i \in X_2 \cup Y_2$ we set $p_i = 0$, for each $i \in X_1$, we set $p_i = B$, and for each $j \in Y_1$ we set $p_j = -s_{j-5k}$. Define the vector of phase angles $\vec{\theta}$ as follows:

$$\theta_i = \begin{cases} B & 1 \leq i \leq k \\ 0 & k+1 \leq i \leq 2k \\ -s_{i-2k}/k & 2k+1 \leq i \leq 5k \\ -s_{i-5k}/k - s_{i-5k} & 5k+1 \leq i \leq 8k \end{cases}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

If $\mathbf{A}$ is the admittance matrix of G, it is easy to check that $\mathbf{A}\vec{\theta} = \vec{p}$. Now define $\vec{\theta'}$ as follows:

$$\theta'_i = \begin{cases} B & 1 \le i \le k \\ 0 & k+1 \le i \le 2k \\ -s_{i-2k} & 2k+1 \le i \le 5k \\ -2s_{i-5k} & 5k+1 \le i \le 8k \end{cases}$$

Now given $\vec{\theta'}_{\bar{H}}$, since each node in H is connected to an exactly one distinct node in $\bar{H}$, there exist a matching between the nodes in H and $\bar{H}$ that covers nodes in H and therefore from [3, Corollary 2], $\vec{\theta'}_{H}$ will be determined uniquely. Hence, we can assume that $\vec{\theta'}$ is given for all the nodes. Now we prove that there exist a set of line failures F in H such that $\mathbf{A}'\vec{\theta'} = \vec{p}$ if, and only if, there exist a solution to the 3-partition problem. Given the way we build the graph G and since the set of failures should be in H, the rest of the proof is exactly similar to the proof of Lemma 2. □

Proof of Lemma 4: Again we reduce the 3-partition partition problem to this problem. The proof is similar to the proof of Lemma 3. Given an instance of a 3-partition problem, we build a graph G, subgraph H, and supply and demand vector $\vec{p}$ exactly as in the proof of Lemma 3. Define $\vec{\theta^\star}_{\bar{H}} = \vec{\theta'}_{\bar{H}}$ as defined in Lemma 3 and $\vec{\theta^\star}_{H} = \vec{z}$, in which $\vec{z}$ is a random vector with arbitrary distribution with no positive probability mass in any proper linear subspace. For any $i \in X_1$, node i is only connected to node $i+k$. Since $\theta_i = B$ and $\theta_{k+i} = z_i$ for a random variable z_i, $\theta_i - \theta_{k+i} \neq B$ almost surely. So in order for the flow equations to hold, either both $i, i+k \in H_0$ or $i+k \in H_0$. The same argument holds for any node $j \in Y_1$ and its only neighbor $j-3k$. So in order for the problem to have a solution, H_0 should contain both X_2 and Y_2. On the other hand, since $|V_{H_0}| \le |V|/2$, therefore $H_0 = G[X_2 \cup Y_2] = H$ is the only possible attacked area. Now since $H_0 = H$, we can assume that the attacked area is given and the rest of the proof is exactly similar to the proof of Lemma 3. □

Proof of Lemma 16: Define $s_k := \sum_{i=1}^{k} w_i$. It is known that

$$f_{s_k}(x) = \frac{\lambda e^{-\lambda x}(\lambda x)^{k-1}}{(k-1)!}.$$

Now since w_is are i.i.d. random variables, $\sum_{i=k+1}^{m} w_i \sim s_{m-k}$. Therefore, all we need to compute is $Pr(s_k < s_{m-k})$.

$$\begin{aligned} Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) &= \int_0^\infty Pr(s_{m-k} - s_k = a)\, da \\ &= \int_0^\infty \int_0^\infty Pr(s_k = y) Pr(s_{m-k} = y+a)\, dy\, da \\ &= \int_0^\infty \int_0^\infty \frac{\lambda e^{-\lambda y}(\lambda y)^{k-1}}{(k-1)!} \frac{\lambda e^{-\lambda(y+a)}(\lambda(y+a))^{m-k-1}}{(m-k-1)!}\, dy\, da \\ &= \int_0^\infty \frac{\lambda^m e^{-2\lambda y} y^{k-1}}{(k-1)!(m-k-1)!} \Big(\int_0^\infty e^{-\lambda a}(y+a)^{(m-k-1)} da \Big) dy. \end{aligned} \tag{9}$$

On the other hand, by defining $z := \lambda(y+a)$, we have:

$$\int_0^\infty e^{-\lambda a}(y+a)^{(m-k-1)} da = \frac{e^{\lambda y}}{\lambda^{m-k}} \int_{\lambda y}^\infty e^{-z} z^{m-k-1}\, dz.$$

Define $T(n+1) := \int_{\lambda y}^\infty e^{-z} z^n\, dz$. Using partial integration:

$$\begin{aligned} T(n+1) &= \big[-e^{-z}z^n\big]_{\lambda y}^\infty + \int_{\lambda y}^\infty n z^{n-1} e^{-z}\, dz \\ &= e^{-\lambda y}(\lambda y)^n + nT(n) = n! e^{-\lambda y} \sum_{i=0}^{n} \frac{(\lambda y)^i}{i!}. \end{aligned}$$

Using equation above in (9) results in:

$$\begin{aligned} &Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) = \\ &= \int_0^\infty \frac{\lambda^n e^{-2\lambda y} y^{k-1}}{(k-1)!(m-k-1)!} \frac{e^{\lambda y}}{\lambda^{m-k}} T(m-k) \\ &= \frac{\lambda^k}{(k-1)!} \int_0^\infty e^{-2\lambda y} y^{k-1} \Big(\sum_{i=0}^{m-k-1} \frac{(\lambda y)^i}{i!} \Big) dy \\ &= \frac{\lambda^k}{(k-1)!} \sum_{i=0}^{m-k-1} \Big(\int_0^\infty e^{-2\lambda y} y^{k-1} \frac{(\lambda y)^i}{i!}\, dy \Big). \end{aligned}$$

By defining $x := 2\lambda y$ and using Gamma function:

$$\begin{aligned} &Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) = \\ &= \frac{\lambda^k}{(k-1)!} \sum_{i=0}^{m-k-1} \Big(\frac{\lambda^{-k}}{i! 2^{i+k}} \int_0^\infty e^{-x} x^{k+i-1}\, dx \Big) \\ &= \frac{\lambda^k}{(k-1)!} \sum_{i=0}^{m-k-1} \Big(\frac{\lambda^{-k}}{i! 2^{i+k}} (k+i-1)! \Big) \\ &= \sum_{i=0}^{m-k-1} 2^{-i-k} \binom{k+i-1}{i} \\ &= 2^{-(m-1)} \sum_{i=0}^{m-k-1} 2^{(m-1)-(i+k)} \binom{k+i-1}{k-1}. \end{aligned} \tag{10}$$

Now notice that $\sum_{i=0}^{m-k-1} 2^{(m-1)-(i+k)} \binom{k+i-1}{i}$ is equal to the total number of subsets of $\{1, \ldots, m-1\}$ with at least k elements. The reason is that this summation is equal to the total number of subsets that contain $k+i$ and exactly $k-1$ elements from $\{1, 2, \ldots, k+i-1\}$. It is easy to verify that by summing this up on i, we count all the subsets of $\{1, \ldots, m-1\}$ with at least k elements. On the other hand, we can count the total number of subsets of $\{1, \ldots, m-1\}$ with at least k elements using the complement rule. The total number of subsets with at least k elements is equal to the total number of subsets minus number of subsets of size 0,1,…,$k-1$. Hence,

$$\sum_{i=0}^{m-k-1} 2^{(m-1)-(i+k)} \binom{k+i-1}{k-1} = 2^{m-1} - \sum_{i=0}^{k-1} \binom{m-1}{j}.$$

Now using the equation above in (10) and using the equality $2^{m-1} = \sum_{i=0}^{m-1} \binom{m-1}{j}$, proves the lemma. □

Proof of Corollary 6: It is easy to see that if $k \le (m-1)/2$, then $\sum_{j=k}^{m-1} \binom{m-1}{j} \ge 2^{m-2}$. Therefore from Lemma 16, $Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) \ge 1/2$ and there is nothing left

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2018.2837894, IEEE Transactions on Network Science and Engineering

to prove. So assume $k = m/2 + \Theta(\sqrt{m})$. It is proved in [44, Lemma 10.8] that for any $1/2 < \alpha < 1$,

$$\frac{2^{n\mathbf{H}(\alpha)}}{\sqrt{8n\alpha(1-\alpha)}} \leq \sum_{j=\alpha n}^{n} \binom{n}{k},$$

in which $\mathbf{H}(\alpha) = -\alpha \log_2(\alpha) - (1-\alpha)\log_2(1-\alpha)$ is the entropy function. Now to prove Corollary 6, select $n = m - 1$, and $\alpha = 1/2 + \epsilon$ for $\epsilon = \Theta(1/\sqrt{n})$. First notice that one can show that the Taylor expansion of the entropy function around $1/2$ can be computed as:

$$\mathbf{H}(\alpha) = 1 - \frac{1}{2\ln 2} \sum_{i=1}^{\infty} \frac{(1-2\alpha)^{2i}}{i(2i-1)}.$$

Using approximation above, it is easy to see that $\mathbf{H}(\alpha) \approx 1 - \Theta(\epsilon^2) = 1 - \Theta(1/n)$. Hence, $2^{n\mathbf{H}(\alpha)} = 2^{n-\Theta(1)}$. On the other hand,

$$\sqrt{8n\alpha(1-\alpha)} = \sqrt{8n(1/2+\epsilon)(1/2-\epsilon)} = \sqrt{8n(1/4-\epsilon^2)}$$
$$= \sqrt{2n - \Theta(1)} \approx \Theta(\sqrt{n}).$$

Hence, by replacing n by $m-1$ and using Lemma 16, one can verify:

$$Pr(\sum_{i=1}^{k} w_i < \sum_{i=k+1}^{m} w_i) = \Omega(\frac{1}{\sqrt{m}}).$$

□

ACKNOWLEDGEMENT

This work was supported in part by DTRA grant HDTRA1-13-1-0021, DARPA RADICS under contract #FA-8750-16-C-0054, funding from the U.S. DOE OE as part of the DOE Grid Modernization Initiative, and NSF under grant CCF-1703925 and CCF-1423100.

REFERENCES

[1] "DARPA Rapid Attack Detection, Isolation and Characterization Systems (RADICS)," http://goo.gl/5Einfw.

[2] "Analysis of the cyber attack on the Ukrainian power grid," Mar. 2016, http://www.nerc.com/pa/CI/ESISAC/Documents/E-ISAC_SANS_Ukraine_DUC_18Mar2016.pdf.

[3] S. Soltan, M. Yannakakis, and G. Zussman, "Power grid state estimation following a joint cyber and physical attack," *to appear in IEEE Trans. Control Netw. Syst. (available on IEEE Xplore Digital Library)*, 2017.

[4] "IEEE benchmark systems," available at http://www.ee.washington.edu/research/pstca/.

[5] R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.

[6] C. Phillips, "The network inhibition problem," in *Proc. ACM STOC'93*, May 1993.

[7] J. Kleinberg, M. Sandler, and A. Slivkins, "Network failure detection and graph connectivity," in *Proc. ACM-SIAM SODA'04*, Jan. 2004.

[8] H. Zhang, Y. Shen, and M. T. Thai, "Robustness of power-law networks: its assessment and optimization," *J. Comb. Opt.*, vol. 32, no. 3, pp. 696–720, 2016.

[9] D. L. Alderson, G. G. Brown, and W. M. Carlyle, "Operational models of infrastructure resilience," *Risk Analysis*, vol. 35, no. 4, pp. 562–586, 2015.

[10] S. Ciavarella, N. Bartolini, H. Khamfroush, and T. La Porta, "Progressive damage assessment and network recovery after massive failures," in *Proc. IEEE INFOCOM'17*, May 2017.

[11] D. Z. Tootaghaj, H. Khamfroush, N. Bartolini, S. Ciavarella, S. Hayes, and T. La Porta, "Network recovery from massive failures under uncertain knowledge of damages," in *Proc. IFIP Networking'17*, June 2017.

[12] T. Nesti, J. Nair, and B. Zwart, "Reliability of DC power grids under uncertainty: a large deviations approach," *arXiv preprint arXiv:1606.02986*, 2016.

[13] A. Pinar, J. Meza, V. Donde, and B. Lesieutre, "Optimization strategies for the vulnerability analysis of the electric power grid," *SIAM J. Optimiz.*, vol. 20, no. 4, pp. 1786–1810, 2010.

[14] T. Kim, S. J. Wright, D. Bienstock, and S. Harnett, "Analyzing vulnerability of power systems with continuous optimization formulations," *IEEE Trans. Net. Sci. Eng.*, vol. 3, no. 3, pp. 132–146, 2016.

[15] A. Bernstein, D. Bienstock, D. Hay, M. Uzunoglu, and G. Zussman, "Power grid vulnerability to geographically correlated failures - analysis and control implications," in *Proc. IEEE INFOCOM'14*, Apr. 2014.

[16] L. Gan, A. Wierman, U. Topcu, N. Chen, and S. H. Low, "Real-time deferrable load control: handling the uncertainties of renewable generation," in *Proc. ACM e-Energy'13*, May 2013.

[17] I. Dobson, B. Carreras, V. Lynch, and D. Newman, "Complex systems analysis of series of blackouts: cascading failure, critical points, and self-organization," *Chaos*, vol. 17, no. 2, p. 026103, 2007.

[18] D. Bienstock, *Electrical Transmission System Cascades and Vulnerability: An Operations Research Viewpoint*. SIAM, 2016.

[19] H. Kesavareddigari, A. Eryilmaz, and R. Srikant, "Controlled link shedding for maximizing supportable demand of a disrupted power network," in *Proc. IEEE CDC'16*, 2016.

[20] H. Cetinay, S. Soltan, F. A. Kuipers, G. Zussman, and P. Van Mieghem, "Comparing the effects of failures in power grids under the ac and dc power flow models," *to appear in IEEE Trans. Netw. Sci. Eng.*, 2017.

[21] J. Kim and L. Tong, "On topology attack of a smart grid: undetectable attacks and countermeasures," *IEEE J. Sel. Areas Commun*, vol. 31, no. 7, pp. 1294–1305, 2013.

[22] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, p. 13, 2011.

[23] G. Dán and H. Sandberg, "Stealth attacks and protection schemes for state estimators in power systems," in *Proc. IEEE SmartGridComm'10*, 2010.

[24] O. Vukovic, K. C. Sou, G. Dán, and H. Sandberg, "Network-layer protection schemes against stealth attacks on state estimators in power systems," in *Proc. IEEE SmartGridComm'11*, 2011.

[25] S. Li, Y. Yılmaz, and X. Wang, "Quickest detection of false data injection attack in wide-area smart grids," *IEEE Trans. Smart Grid*, vol. 6, no. 6, pp. 2725–2735, 2015.

[26] J. Kim, L. Tong, and R. J. Thomas, "Subspace methods for data attack on state estimation: A data driven approach," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1102–1114, 2015.

[27] R. B. Bobba, K. M. Rogers, Q. Wang, H. Khurana, K. Nahrstedt, and T. J. Overbye, "Detecting false data injection attacks on dc state estimation," in *Proc. SCS CPSWEEK'10*, vol. 2010, 2010.

[28] A. Teixeira, S. Amin, H. Sandberg, K. H. Johansson, and S. S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *Proc. IEEE CDC'10*, 2010.

[29] J. E. Tate and T. J. Overbye, "Line outage detection using phasor angle measurements," *IEEE Trans. Power Syst.*, vol. 23, no. 4, pp. 1644–1652, 2008.

[30] ——, "Double line outage detection using phasor angle measurements," in *Proc. IEEE PES'09*, July 2009.

[31] M. Garcia, T. Catanach, S. Vander Wiel, R. Bent, and E. Lawrence, "Line outage localization using phasor measurement data in transient state," *IEEE Trans. Power Syst.*, vol. 31, no. 4, pp. 3019–3027, 2016.

[32] H. Zhu and G. B. Giannakis, "Sparse overcomplete representations for efficient identification of power line outages," *IEEE Trans. Power Syst.*, vol. 27, no. 4, pp. 2215–2224, 2012.

[33] S. Soltan and G. Zussman, "Power grid state estimation after a cyber-physical attack under the AC power flow model," in *Proc. IEEE PES-GM'17*, 2017.

[34] ——, "EXPOSE the line failures following a cyber-physical attack on the power grid," *arXiv preprint arXiv:1709.07399*, Sept. 2017.

[35] Z. Li, M. Shahidehpour, A. Alabdulwahab, and A. Abusorrah, "Bilevel model for analyzing coordinated cyber-physical attacks

on power systems," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2260–2272, 2016.
[36] R. Deng, P. Zhuang, and H. Liang, "CCPA: Coordinated cyber-physical attacks and countermeasures in smart grid," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2420–2430, 2017.
[37] J. Zhang and L. Sankar, "Physical system consequences of unobservable state-and-topology cyber-physical attacks," *IEEE Trans. Smart Grid*, vol. 7, no. 4, 2016.
[38] D. Bienstock and M. Escobar, "Computing undetectable attacks on power grids," *ACM SIGMETRICS Performance Evaluation Review*, vol. 45, no. 2, pp. 115–118, 2017.
[39] J. A. Bondy and U. Murty, "Graph theory, volume 244 of graduate texts in mathematics," 2008.
[40] M. R. Garey and D. S. Johnson, "Computers and intractability: a guide to the theory of NP-completeness," 1979.
[41] R. Bapat, *Graphs and matrices*. Springer, 2010.
[42] Y. Zhao, A. Goldsmith, and H. V. Poor, "On PMU location selection for line outage detection in wide-area transmission networks," in *Proc. IEEE PES'12*, July 2012.
[43] H. Zhu and Y. Shi, "Phasor measurement unit placement for identifying power line outages in wide-area transmission system monitoring," in *HICSS'14*, 2014, pp. 2483–2492.
[44] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. Elsevier, 1977.

Saleh Soltan is a postdoctoral research associate in the department of Electrical Engineering at Princeton University. In 2017, he obtained the Ph.D. degree in Electrical Engineering from Columbia University. He received B.S. degrees in Electrical Engineering and Mathematics (double major) from Sharif University of Technology, Iran in 2011 and the M.S. degree in Electrical Engineering from Columbia University in 2012. He is the Gold Medalist of the 23rd National Mathematics Olympiad in Iran in 2005 and the recipient of Columbia University Electrical Engineering Armstrong Memorial Award in 2012.

Mihalis Yannakakis is the Percy K. and Vida L. W. Hudson Professor of Computer Science at Columbia University. Prior to joining Columbia, he was Head of the Computing Principles Research Department at Bell Labs and at Avaya Labs, and Professor of Computer Science at Stanford University. Dr. Yannakakis received his PhD from Princeton University. He has served on the editorial boards of several journals, including as editor-in-chief of the SIAM Journal on Computing, and has chaired various conferences, including the IEEE Symposium on Foundations of Computer Science, the ACM Symposium on Theory of Computing, and the ACM Symposium on Principles of Database Systems. Dr. Yannakakis is a recipient of the Knuth Prize, a member of the National Academy of Engineering, of Academia Europaea, a Fellow of the ACM, and a Bell Labs Fellow.

Gil Zussman received the Ph.D. degree in electrical engineering from the Technion in 2004 and was a postdoctoral associate at MIT in 2004-2007. He is currently an Associate Professor of Electrical Engineering at Columbia University. He is a co-recipient of 7 paper awards including the ACM SIGMETRICS06 Best Paper Award, the 2011 IEEE Communications Society Award for Advances in Communication, and the ACM CoNEXT'16 Best Paper Award. He received the Fulbright Fellowship, the DTRA Young Investigator Award, and the NSF CAREER Award, and was a member of a team that won first place in the 2009 Vodafone Foundation Wireless Innovation Project competition.