Bayesian Network-based Reasoning for Assisting the Differentiation between Cyberattacks and Undetected Faults in DER Systems

Mohammad Shamim Ahsan
College of IST
The Pennsylvania State University
University Park, PA, USA
msa6152@psu.edu

Minghui Zhu
Dept. of Electrical Engineering
The Pennsylvania State University
University Park, PA, USA
muz16@psu.edu

Peng Liu
College of IST
The Pennsylvania State University
University Park, PA, USA
pxl20@psu.edu

***Abstract*—In the previous work, we proposed a combined analysis integrating cyberspace information and physical side measurements to distinguish cyberattacks and physical faults in DER systems. However, the approach is not scalable and resource-intensive because of the reliance on manual reasoning performed by human analysts. This paper aims to address this limitation by introducing a novel Bayesian Network (BN)-based reasoning technique. In particular, the construction of the network follows an automatic and generic strategy such that the structure and conditional probability tables (CPTs) can capture domain-specific knowledge of DER systems. To evaluate the effectiveness of the proposed reasoning technique, we have conducted four case studies, which demonstrate the correctness of the approach and its ability to handle a substantial amount of real-world uncertainties. However, further investigation reveals that BN models are sensitive to small changes in the CPT parameters and evidences, implying their potential drawbacks in certain scenarios.**

***Index Terms*—Bayesian Network, Reasoning, Cyberattacks, Physical Faults, DER System**

I. Introduction

Despite the fact that a substantial amount of progress has been made during the past two decades in the area of power systems fault detection and localization, not all faults can be detected: various faults (e.g., faults under low irradiation conditions and simultaneous faults in PV arrays [1]) remain undetected in DER systems. Accordingly, the differentiation between undetected faults and cyberattacks becomes an important open problem in DER systems. Because both undetected faults and cyberattacks can trigger anomaly detection systems (ADSs) to trigger indistinguishable alerts, ADSs (e.g., [2], [3]) cannot achieve differentiation. The presence of an alert only tells something is abnormal but not revealing the exact root cause. Although there are existing works (e.g., [4]–[6]) aiming to achieve differentiation, they suffer from limited practicality: they either rely on physical measurements (which are not always available) or assume that the physical laws can reliably handle real-world signal noise. In our previous work (refer to §II for details), we aimed to address these limitations by proposing a combined analysis technique leveraging the complementary nature of the information held in cyberspace logs and physical measurements. In particular, a non-trivial taint-styled dependency graph was constructed to capture the mapping relation between program variables and physical-side measurements, as well as to trace data dependencies within DERMS (DER management system) application program. Subsequently, a set of domain-specific patterns was derived and matched (against the dependency graph using two widely-used cyberspace logs and physical measurements) to identify the root cause of an anomaly detected by ADSs.

Although the combined analysis can address the aforementioned limitations of the existing differentiation approaches, the *scalability* issue remains with the *manual* reasoning carried out by human analysts. Specifically, this process is costly because of its heavy reliance on human resources. In addition, as the number of pattern-matching results and real-world uncertainties increase, the reasoning process can become not only time-consuming but also error-prone.

Our Contributions. To address these challenges, in this work, we introduce a Bayesian Network (BN)-based reasoning approach. The rationale for employing a BN is as follows. First, the proposed network construction is *automatic and generic*, capable of handling any number of pattern-matching results, thereby reducing the reasoning time to some extent. Second, according to the BN principle, the use of conditional probability distributions can handle real-world uncertainties, especially possible inaccuracies in pattern matching results arising from – (1) *imperfect logs*: fluctuation is often observed in cyberspace logs (e.g., network traffic packets or application log) and field (e.g., PMU) measurements, and (2) *insufficient data sources*: in certain cases (e.g., program variables involved in intermediate mathematical computations), where logs are not always available in practice.

Specific to the structure, the proposed Bayesian network captures the distinct characteristics of the DER system domain. In addition, the generated conditional probability tables (CPTs) integrate *domain-specific* causal relationships among interconnected nodes, including a *special* relationship derived from the taint-styled dependency graph. In this way, the reasoning becomes not only automatic but also incorporates domain

979-8-3315-8138-1

knowledge. Finally, the efficacy of the proposed approach is demonstrated through four case studies, followed by a *sensitivity analysis* and a *what-if evidence analysis* to assess potential pitfalls in Bayesian reasoning.

II. Preliminaries

The previous work introduced a novel approach to automatically synthesizing physical side and cyberspace findings, for the first time to date, differentiating physical faults and cyberattacks in DER systems. Specifically, the approach was based on three key observations.

Observation 1. Mapping Relationships. We found some crucial mappings between variables from which physical measurements are collected and the program variables used by DERMS applications, as well as between dispatch-related program variables and the physical side control of the DER. These mapping relationships can provide a fundamental reason why combined analysis could create a unique capability.

Observation 2. Data Dependency and Taint Analysis. Program variables influenced by the mappings can be tracked using *taint analysis* [7], [8] – a program analysis technique used to trace information flow within an application program coming from some untrusted sources (e.g., API functions, user inputs), which propagates along the program execution path and ends up at some release points or sensitive functions (known as *sinks*). Data entered into the program by these sources is considered *tainted*, which, during propagation, makes any dependent variables tainted as well. Regarding the mappings, if taint propagation is observed in cyberspace from a variable that is mapped to a physical side measurement to a variable that is mapped to a physical side control, then the extended cyber-physical taint propagation would provide a unique lens to examine not only the (varied) effects of different cyberattacks, but also the differences between undetected faults and cyberattacks. Although this kind of mapping is highly desirable, it is non-trivial to achieve since existing automatic taint analysis techniques fall short of meeting the synthesized analysis requirements of a DER system.

Observation 3. Combined Analysis. Currently, the connections between physical measurements and cyberspace information items (e.g., data dependencies between program variables) are not being automatically analyzed in real-world practice. In fact, significant semantic gaps exist between physical measurements and cyberspace information items.

Based on these observations, first, a special kind of data dependency graph was constructed using a novel virtual physical variable-oriented taint analysis (PVOTA) algorithm. Formally, a *data dependency graph* (or simply a *dependency graph*) is defined as a directed graph that begins with taint source nodes, connects dependent code statements and variables through direct edges, and ends at sink nodes. Then, we simplified the graph using an innovative node pruning technique based on a set of context-dependent operations. Next, a set of patterns that capture domain-specific knowledge was derived to bridge the semantic gaps between the cyber and physical sides. These patterns utilize three types of information: (i) data files holding PMU measurements (referred to as field measurements), (ii) Network packets PCAP (packet capture) files used to monitor network activities, and (iii) DERMS application log generated during the execution of the application program.

In a DER system, a data dependency graph typically consists of three types of nodes: response message-related, dispatch message-related, and other program variables-related. However, for only response and dispatch messages, corresponding field measurements and network packets exist. In accordance with this knowledge, a set of patterns, referred to as *global patterns*, is devised to match abnormal system-wide event sequences. On the other hand, for other program variables that are only recorded in the application log, a separate set of patterns, called *local patterns*, is used to track their values. The derived global and local patterns are $GPTN_1 - GPTN_8$ and $LPTN_1 - LPTN_3$, respectively. $GPTN_1$, $GPTN_2$, $GPTN_7$, and $GPTN_8$ represent inconsistency in network packets; $GPTN_3$ and $GPTN_4$ represent inconsistency in DERMS application log; whereas, $GPTN_5$ and $GPTN_6$ represent inconsistency in PMU measurements. Regarding the local patterns, $LPTN_1$ denotes expected deviation in a node; whereas, $LPTN_2$ and $LPTN_3$ represent unexpected deviations.

In the end, the efficacy of the proposed automatic integrated analysis was evaluated through four case studies covering failure incidents caused by the system fault (Case 1), FDI attack (Case 2), and memory corruption attacks (Cases 3 and 4). For each failure incident, a sequence of events was extracted from the corresponding log files. At the beginning of the analysis, the analyst utilized this sequence to find the specific tainted path(s) in the dependency graph ignoring the unrelated ones. After that, the analyst went through every node in that path and searched for matched patterns. Finally, a decision on the root cause was made based on the pattern-matching results (see Table I).

TABLE I
Pattern Matching Results of Case Studies

Case #	Pattern Matching Results
1	$GPTN_5$ found; $GPTN_1$ or $GPTN_2$, $GPTN_3$ or $GPTN_4$, $GPTN_7$ or $GPTN_8$ not found; $LPTN_1$ were found in some response and dispatch related nodes; no $LPTN_2$ and $LPTN_3$ were found
2	$GPTN_1$ found; $GPTN_3$ or $GPTN_4$, $GPTN_5$ or $GPTN_6$, and $GPTN_7$ or $GPTN_8$ not found; $LPTN_1$ were found in some response and dispatch related nodes; no $LPTN_2$ and $LPTN_3$ were found
3	$GPTN_4$ found; $GPTN_1$ or $GPTN_2$, $GPTN_5$ or $GPTN_6$, and $GPTN_7$ or $GPTN_8$ not found; $LPTN_1$ found in some response-related nodes; one unexpected $LPTN_2$ and multiple unexpected $LPTN_3$ found (dispatch related)
4	$GPTN_5$ found; $GPTN_1$ or $GPTN_2$, $GPTN_3$ or $GPTN_4$, and $GPTN_7$ or $GPTN_8$ not found; $LPTN_1$ found in response-related nodes; one $LPTN_2$ and multiple $LPTN_3$ found (dispatch related, unexpected)

III. Problem Statement

How can the reasoning process be automated to enable decision-making on the root causes without human intervention? In previous work, human analysts used pattern-matching results to perform manual reasoning based on domain knowledge and human cognition. However, the process is

notably time-consuming and error-prone, as it relies on human judgment and manual analysis. To address this problem, an alternate solution is essential that can not only perform automatic reasoning and incorporate domain-specific knowledge but also handle real-world uncertainties.

IV. Proposed Approach

A. Bayes Network Construction

A Bayesian Network (BN) implements a graphical model structure called a directed acyclic graph, which is defined by a set of nodes and a set of directed edges. The nodes represent random variables, and the edges represent direct dependencies among the variables. In this work, we propose a BN model that integrates *domain-specific* knowledge to capture causal relationships between different nodes. The network contains three groups of nodes: (Group-1) patterns matching results nodes, (Group-2) deviation nodes, and (Group-3) root cause nodes. According to the case studies conducted in the previous work (see Section II), it is observed that although the co-existence of global and local patterns is necessary, each type of pattern has their own significance. Specifically, the global patterns can effectively capture distinct characteristics of field-level and network-level manipulations. In contrast, the local patterns are indispensable to analyze deviation in the application program as well as to make an appropriate decision, especially when the matched global pattern appears to imply a conflicting outcome (referring to Case 4). Based on these observations, the proposed Bayesian Network is divided into two subgraphs: (i) *global subgraph*, which includes global pattern matching results and their dependent nodes, and (ii) *local subgraph*, consists of local pattern matching results and their dependent nodes.

1) Global Subgraph: The motivation of the global subgraph is to capture the causal relationships between the root cause and deviations at network and field levels. Since three types of information - (i) data files holding PMU measurements, (ii) Network packet files monitoring network activities, and (iii) DERMS application log - are utilized to obtain pattern matching results (Section II), the global subgraph is divided into three smaller portions containing PMU measurements, network, and DERMS application-related nodes.

Nodes. The first category (Group-1) of nodes (a.k.a, prior nodes) in the subgraph simply contains pattern-matching results retrieved from the previous work, while the intermediate (Group-2) nodes represent the state of deviation in the corresponding portions. From Figure 1, 'P1'-'P4' are Group-1 nodes; 'I1'-'I5', and 'I9' are Group-2 nodes of the global subgraph. It should be noted that the last group (Group-3) of nodes, which represents the three root causes - system fault ('R1'), false data injection ('R2'), and memory corruption ('R3') - does not belong to either the global or the local subgraph, as their probabilities are aggregated from both.

Edges. The edges connecting Group-1 and Group-2 nodes indicate the conditional probabilities of deviation in the corresponding regions. For example, the edge between nodes 'P3' and 'I4' represents how the presence of global pattern $GPTN_1$ or $GPTN_7$ influences the existence or non-existence of deviation in the program side. However, in several cases, edges are used among Group-2 nodes located at different layers to represent the aggregated effect of the deviations in the global region. Regarding the edges among Group-2 and Group-3 nodes, two incoming edges can be observed in each of the Group-3 (root causes) node. These edges represent the conditional probability of a specific root cause (posteriors) measured according to the degree of global and local regions.

2) Local Subgraph: The local subgraph contains the pattern matching results observed during DERMS application program analysis. The typical program structure consists of three main phases: first, retrieving physical-layer measurements through an API interface; second, performing computations to derive updated measurement values for the targeted equipments; and finally, dispatching updates to the environment to process or control DER system. Based on our prior findings, only response- and dispatch-related patterns are observable in real-world settings due to the unavailability or insufficiency of data sources associated with intermediate mathematical operations. In accordance with it, we partition the local subgraph into two portions: response-related nodes and dispatch-related nodes. In both cases, Group-1 nodes represent the pattern matching results of the corresponding partitions, and Group-2 nodes represent the states of the related deviations, which are eventually aggregated as the "`Degree of Local Deviation`" (node 'I8'). Specifically, according to Figure 1, nodes 'P5'-'P13' belong to Group-1 and 'I6'-'I8' belong to Group-2 of the local subgraph. Like in the global subgraph, the edges connecting local subgraph nodes represent their causal relationships.

In previous work, we devised a pattern matching algorithm leveraging a novel taint analysis technique to capture data dependency between program variables in DERMS application programs. By nature of these dependencies, they have a significant influence on the patterns *expected* to match in the corresponding variables. Based on this insight, we utilize these data dependencies to integrate *special* domain-specific causal relationships in the local subgraph, since only this portion handles application program-related nodes. In particular, Group-1 nodes (i.e., 'P5'-'P9' and 'P10'-'P13') are connected with directed edges, where each edge indicates a data dependency. For example, in Figure 1, the edge from node 'P5' to node 'P6' represents a data dependency between the program variables `measurements` and `angle` in the taint-styled dependency graph. However, it is important to note that the Group-1 nodes in the BN model and the corresponding nodes in the taint analysis are different in principle. Particularly, in the taint graph, nodes denote program variables or code statements; whereas, Group-1 nodes in the local subgraph represent the pattern matching results in the corresponding nodes of the taint graph. Because the model captures taint-style data dependencies in an *indirect* manner, it is characterized as *passive*.

Definitions. In Bayesian inference, if the variable represented

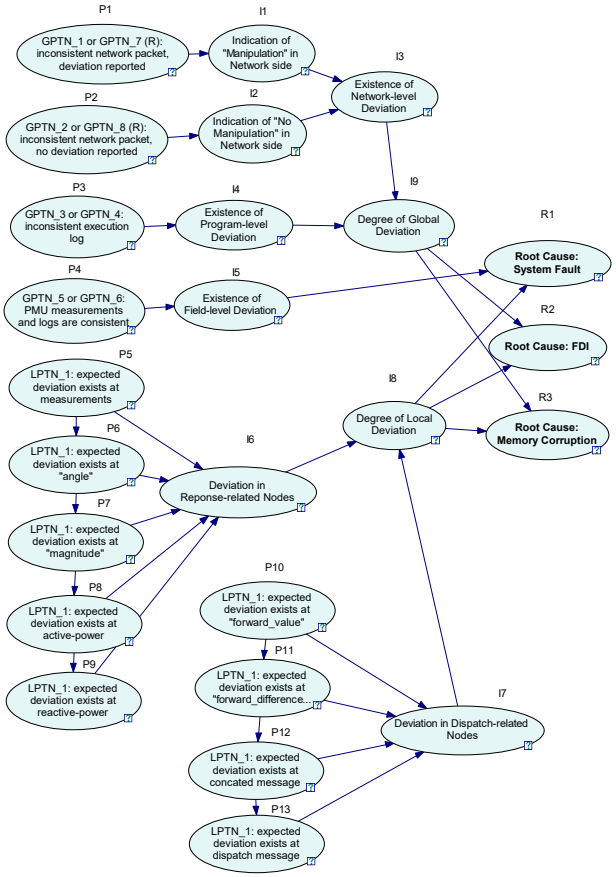
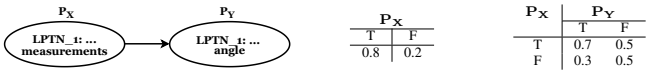


Fig. 1. Bayesian network model for Case Studies 1 and 2

by a node is observed, then the node is said to be an *evidence*. In addition, the node is called *hard evidence* if all but one state are assigned a zero probability; otherwise, it is called *soft evidence*

B. CPT Generation

To estimate the parameters of conditional probability tables (CPTs), we employ *expert elicitation*, a widely accepted methodology in the cybersecurity domain [9], [10]. As an illustrative example, the CPTs for the Group-2 nodes in the global subgraph are parameterized according to the state of the global pattern. Specifically, if the global pattern is observed (evidence state: `Found`), the deviation states follow the distribution: `Yes`: 0.9 and `No`: 0.1. If the pattern is not observed (evidence state: `Not Found`), the deviation probabilities are `Yes`: 0.2 and `No`: 0.8. Although all CPTs are generated following a similar approach, it is impractical to present each one in detail. However, special attention is given to the CPTs associated with the local subgraph nodes, capturing taint-styled data dependencies. For example, if node P_X is the first Group-1 node in the local subgraph, its prior probabilities are given as follows: $P(X = T) = 0.8$ and $P(X = F) = 0.2$. In contrast, the conditional probabilities of the dependent Group-1 node P_Y are $P(X = T, Y = T) = 0.7, P(X = T, Y = F) = 0.3, P(X = F, Y = T) = 0.5, P(X = F, Y = F) = 0.5$. The beliefs of the consecutive dependent nodes also follow the same rule.



P_X	
T	F
0.8	0.2

P_X	P_Y	
	T	F
T	0.7	0.5
F	0.3	0.5

Fig. 2. CPT generation in the PDDM

V. Simulation Results

A. Case Studies

We construct four BN models for the previous case studies, using the corresponding pattern-matching results. The GeNIe Modeler[1], a graphical user interface (GUI) tool, is used to construct the networks.

Case 1. This BN consists of thirteen Group-1 nodes (pattern matching results), nine Group-2 (intermediate) nodes, and three Group-3 (root causes) nodes. Among the Group-1 nodes, four are global pattern-matching results, five are response-related, and four are dispatch-related pattern-matching results nodes. According to the special causal relationship adopted from the taint-styled dependency graph, five Group-2 response-related nodes and four Group-2 dispatch-related nodes are connected with directed edges, each edge denotes the data dependency between two consecutive nodes. Finally, three nodes are utilized for three root causes, especially system fault, FDI, and memory corruption. For the first case study, the measured posterior probabilities of root causes are: system fault: 78%, FDI: 55%, and memory corruption: 31%, reflecting the correctness of the BN reasoning, given that the ground truth is a system fault.

Case 2. Although the network is similar to the previous one, the evidence differs slightly, as the global pattern $GPTN_1$, rather than $GPTN_5$, is observed during the failure incident. Consequently, the probabilities of the root causes are system fault: 25%, FDI: 68%, and memory corruption: 25%. This result also demonstrates the efficacy of the proposed BN reasoning.

Case 3. The structure of this Bayes network differs only in the dispatch-related portion of the local subgraph. The data dependency model consists of four nodes: one $LPTN_2$ and three $LPTN_3$ nodes. Applying the pattern matching results as *hard evidence*, the beliefs of the root causes are calculated as follows: system fault: 19%, FDI: 29%, and memory corruption: 75%.

Case 4. The BN of this case study is identical to the previous one, except the global pattern is now $GPTN_5$, indicating the possibility of a system fault. However, the final reasoning associated with root causes is – system fault: 50%, FDI: 24%, and memory corruption: 69%, which is correct, as the ground truth was memory corruption.

B. Sensitivity Analysis

We use the *direct* approach to observe the model's sensitivity to changes in CPT parameters, where parameters

[1]https://support.bayesfusion.com/docs/GeNIe.pdf

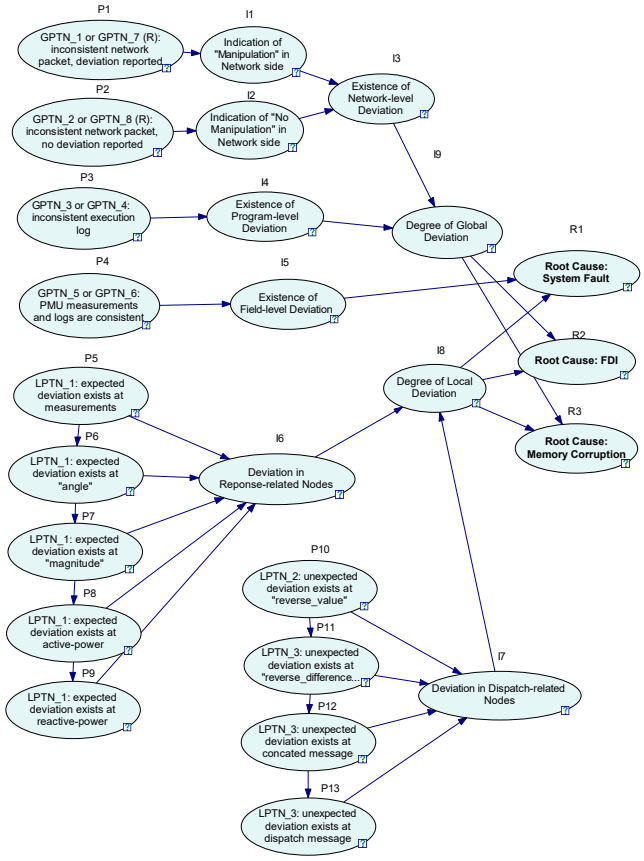


Fig. 3. Bayesian network model for last two case studies

are changed within defined limits based on domain-specific knowledge, and the recomputed outputs are compared to the original outputs [10]. For the first case study, we change the CPT parameters of three nodes: "I5" (Existence of Field-level Deviation), "I8" (Degree of Local Deviation), and "R1" (Root Cause: System Fault). First, for the "I5" node, the parameters of the states `Yes` and `No` are changed from 0.9-0.1 to 0.7-0.3, when $GPTN_5$ is `Found`. Secondly, the conditional probability of the `Expected` state of node "I8" is decreased to 0.8 when "No" deviation is found in the response-related and dispatch-related nodes. Finally, the parameters of the states `Yes` and `No` of node "R1" are changed from 0.9 to 0.7. As a consequence, the posterior probabilities of the root causes become - system fault: 51%, FDI: 51%, and memory corruption: 35%. For the second case study, the CPT parameters of two nodes "R2" (Root Cause: FDI) and "I8" are changed between 0.7 and 0.8 range, which results in a drastic failure in the BN reasoning with probabilities of system fault: 21%, FDI: 52%, and memory corruption: 37%. The results of these two case studies demonstrate that the BN reasoning is sensitive to slight changes, even only in a few CPT tables.

C. What if Evidence Analysis

In realistic settings, situation may arise when one or a few evidences can be missing or not observed. In that case, a BN model can be susceptible to small changes in evidences. Based on this observation, we make one or two nodes as *soft evidence* to examine how the model behaves in realistic settings. For the first case study, making only one node, "$GPTN_5$ `or` $GPTN_6$`: PMU measurements and logs are consistent`" as soft evident with probabilities 0.5-0.5 result in drastic performance failure in the reasoning, as probabilities become – system fault: 52%, FDI: 55%, and memory corruption: 31%. Similarly, in the second study, making the probabilities of one more node "$GPTN_1$ `or` $GPTN_7$ `(R): inconsistent network packet, deviation reported`" to 0.5-0.5 (soft evidence), the posterior probabilities become system fault: 52%, FDI: 61%, and memory corruption: 28%.

VI. Conclusion

This paper presents a Bayesian Network-based approach for automatic reasoning to address the existing limitations of human judgment-based manual reasoning. Through a set of distinct case studies, we have demonstrated that the BN-based technique is effective enough to handle real-world uncertainties. However, further analysis shows that it is susceptible to minor statistical fluctuations in a small subset of CPT parameters and prior probabilities.

References

[1] Y. Zhao, B. Lehman, J.-F. de Palma, J. Mosesian, and R. Lyons, "Fault analysis in solar pv arrays under: Low irradiance conditions and reverse connections," in *2011 37th IEEE Photovoltaic Specialists Conference*. IEEE, 2011, pp. 002 000–002 005.

[2] M. Benninger, M. Hofmann, and M. Liebschner, "Anomaly detection by comparing photovoltaic systems with machine learning methods," in *NEIS 2020; Conference on Sustainable Energy Supply and Energy Storage Systems*. VDE, 2020, pp. 1–6.

[3] Q. Wang, K. Paynabar, and M. Pacella, "Online automatic anomaly detection for photovoltaic systems using thermography imaging and low rank matrix decomposition," *Journal of quality technology*, vol. 54, no. 5, pp. 503–516, 2022.

[4] K. Gupta, S. Sahoo, R. Mohanty, B. K. Panigrahi, and F. Blaabjerg, "Distinguishing between cyber attacks and faults in power electronic systems—a noninvasive approach," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 11, no. 2, pp. 1578–1588, 2022.

[5] S. Sahoo, Y. Yang, and F. Blaabjerg, "Resilient synchronization strategy for ac microgrids under cyber attacks," *IEEE Transactions on Power Electronics*, vol. 36, no. 1, pp. 73–77, 2020.

[6] Y. Shen and Z. Qin, "Detection, differentiation and localization of replay attack and false data injection attack based on random matrix," *Scientific Reports*, vol. 14, no. 1, p. 2758, 2024.

[7] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," *ACM sigplan notices*, vol. 49, no. 6, pp. 259–269, 2014.

[8] M. Sridharan, S. Artzi, M. Pistoia, S. Guarnieri, O. Tripp, and R. Berg, "F4f: taint analysis of framework-based web applications," in *Proceedings of the 2011 ACM international conference on Object oriented programming systems languages and applications*, 2011, pp. 1053–1068.

[9] P. Xie, J. H. Li, X. Ou, P. Liu, and R. Levy, "Using Bayesian networks for cyber security analysis," in *2010 IEEE/IFIP international conference on dependable systems & networks (DSN)*. IEEE, 2010, pp. 211–220.

[10] K. B. Laskey, "Sensitivity analysis for probability assessments in bayesian networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 6, pp. 901–909, 2002.