Research Article

A Minimum Defense Cost Calculation Method for Attack Defense Trees

Bingfeng Xu,¹ Zhicheng Zhong,¹ and Gaofeng He^{2,3}

¹College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China

²School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

³Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing, China

Correspondence should be addressed to Gaofeng He; hegaofeng@njupt.edu.cn

Received 6 April 2020; Revised 29 June 2020; Accepted 10 July 2020; Published 1 August 2020

Academic Editor: Mamoun Alazab

Copyright © 2020 Bingfeng Xu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The cyberphysical system (CPS) is becoming the infrastructure of society. Unfortunately, the CPS is vulnerable to cyberattacks, which may cause environmental pollution, property losses, and even casualties. Furthermore, in contrast to the conventional Internet, the devices in CPSs are more specific, and the device systems may not be upgraded or installed with new programs during their life spans. The selection of the best defense nodes for defeating cyberattacks is quite challenging in CPSs. To overcome this issue, several attack-defense modeled methods have been proposed. However, few existing studies have considered the defense cost, which is usually a determinant in practice. In this paper, we propose a method for choosing optimal defense nodes that (1) can defeat specific attacks and (2) are inexpensive. First, the atom attack defense tree (A2DTree) is proposed by adding constraints to the conventional attack defense tree (ADTree). Second, the algebraic method is used to efficiently calculate the minimum defense cost. On this basis, a minimum defense cost calculation tool is designed and implemented. Finally, the effectiveness of the proposed method is verified with two typical case studies, and a comparative experiment of related work is carried out. The results show that the method can correctly and efficiently identify the optimal defense nodes and calculate the minimum defense cost of a CPS.

1. Introduction

Cyberphysical systems (CPSs) are complex systems that use modern sensor, computing, and network technologies to achieve computation, communication, and control (3C) integration. In recent years, the CPS has been widely recognized as the core technology for promoting the development of Industry 4.0, and it has been successfully applied to control systems in industries such as electricity, medical treatment, transportation, water supply, and natural gas [1, 2]. Hence, the CPS is currently a research area of increased interest in industry and academia [3]. However, since information technologies are deeply used and the communications between various components are mainly achieved through a network, CPSs are vulnerable to cyberattacks [4–6]. Furthermore, due to the high coupling

between physical and network components in CPSs, cyberattacks can trigger physical component failures that have severe consequences, such as environmental pollution, property losses, and even casualties [7, 8]. For example, in 2015, the Ukrainian power network suffered a spear-phishing attack [9]. Hackers used Microsoft Office files containing malicious macros as the attack vector to clear supervisory control and data acquisition (SCADA) system data, resulting in approximately 700,000 residential users in western Ukraine losing power for hours.

To prevent various cyberattacks, it is necessary to adopt strategies for securing CPSs. For instance, each device in the CPS should be installed with an antivirus program, firewall, or intrusion detection system (IDS), or the devices should be frequently upgraded to fix program bugs, as in the Internet. However, in the CPS, there are devices that cannot be

upgraded or cannot be installed with extra programs. For example, in the power industry control system, it is impossible to update the equipment in remote areas in a timely manner [10]. Moreover, during the installation of Industry 4.0 software, it was found that many new software programs could not be installed on older devices [11]. Therefore, selecting the appropriate devices for additional security measures and ensuring that the whole system is secure are vital and challenging issues in CPS research.

To solve these issues, the common approach is to first express the behaviors of attackers and defenders in the CPS by graph models such as attack defense tree (ADTree) [12]. Then, appropriate defense strategies can be selected through cut-set analysis, game theory, and other methods. For example, Shamel-Sendi et al. [13] used ADTree to establish a security model and then proposed a dynamic defense framework that selects an optimal countermeasure by considering the security benefit and attack damage cost. Wang and Liu [14] established a systematic attack defense game model based on the return on attack (ROA) and return on investment (ROI) of an ADTree. Namely, the optimal defense devices are chosen based on ROA and ROI. Chakraborty and Kalaimannan [15] modeled the risk of the CPS in a smart grid and assigned a cost factor to each atom attack node. On this basis, the algebraic method is used to analyze the minimum number of attack nodes that should be defended to prevent the attack target from occurring. In their method, the optimal defense strategy is determined by aiming to defend the fewest attack nodes.

Existing studies mainly design methods for choosing CPS defense nodes by defending the minimum number of attack nodes or preventing an attack with a high defense success rate, without considering the defense cost. However, the defense cost is an important issue that must be considered in practical systems. For example, for the ADTree of a small network system with 15 attack nodes, where the average defense cost for each attack node is \$20,000 [16], if we defend all attack nodes, then the defense cost can reach \$300,000, which is a heavy burden for small-sized and midsized enterprises. Therefore, in actual industrial control systems, cost is an important reference index when developing defensive strategies.

In this paper, we propose a method for choosing CPS defense devices by considering the defense cost. Our goal is to simultaneously prevent a complete attack and minimize the defense cost. To this end, we design a new atom attack defense tree (A2DTree) for modeling attack and defense behaviors in CPS. A2DTree is a variation of ADTree, in which only atom attack nodes have defense strategies and all defense strategies (device information is included in the strategy) are on the leaf nodes, which can help us to identify all potential practical defense devices efficiently and effectively. Since ADTree is commonly used in industry, we also proposed an algorithm that can automatically transform ADTree models into A2DTree models. Based on A2DTree modeling, an efficient calculation for the minimum defense cost of the CPS can be achieved. In this study, we use algebraic methods to avoid traversing all tree nodes recursively, which can significantly speed up the calculation

process. The calculation results are the candidates of optimal defense devices. These devices should be protected so that the final assets can be guaranteed, and the defense cost to the user should also be minimized.

The main contributions of this work are as follows:

- (1) We introduce the problem of selecting CPS defense nodes while considering the cost. The cost can include hardware equipment, software development, labor, and time costs, which should be particularly considered in real-world security defense systems.
- (2) To improve the efficiency of the minimum defense cost calculation, we propose an A2DTree model in which all the defense nodes are leaf nodes. A2DTrees can be directly used to model system security scenarios. Furthermore, we can automatically obtain A2DTree models from conventional ADTree models by moving all the defense nodes to the leaf nodes. We have proven that A2DTree is equivalent to ADTree in terms of the minimum cost calculation.
- (3) Based on the A2DTree model, we design a minimum defense cost calculation algorithm and implement an open-source computing tool. The open-source tool can be downloaded from https://github.com/zzc1/ADTree_Min_DefCost/releases/tag/1.0.

The subsequent chapters of this paper are arranged as follows. Section 2 introduces the work related to the CPS risk model and the quantitative analysis of ADTree. Section 3 provides the method overview. Section 4 presents the proposed A2DTree, provides the algorithm for transforming ADTree to A2DTree, and proves the equivalence of these two models in terms of the minimum defense cost calculation. Section 5 introduces the minimum defense cost calculation algorithm and the complete calculation process. Section 6 illustrates the effectiveness of our method through two typical case studies and demonstrates the efficiency of our method through a comparison with related work. The paper closes with a summary and a discussion of future work.

2. Related Work

To complete the analysis of the CPS defense cost, CPS system security modeling needs to be performed first. Currently, the CPS security modeling methods commonly used in industry mainly include graphical models, such as attack tree (ATree) [17], ADTree [12], and attack graph [18]. Among them, ATree is a systematic attack scenario modeling method proposed by Schneier [19] and formally defined by Mauw and Oostdijk [20], and it is widely used in system security assessment. The ATree model attacks scenarios layer by layer from top to bottom and decomposes attack targets into atom attacks layer by layer. On this basis, the attack scenario can be qualitatively and quantitatively analyzed. However, the ATree can describe only the attack scenario and cannot represent the interaction between the attacker and the defender. To this end, Kordy et al. [12] proposed ADTree based on ATree by adding defense nodes to ATree. ADTree can model attack defense scenarios and perform security

assessments of the system [21], thus enabling the attack defense cost characteristics of a CPS to be analyzed.

Since ADTree is a semiformal model, a common method is to establish its analysis method based on formal methods. For example, the authors of [22] classified the quantitative analysis problems of ADTrees and examined the application of formal methods for establishing ADTree analysis methods. Jhawar et al. [23] adopted the continuous-time Markov chain (CTMC) for determining the quantitative analysis semantics of the ADTree. They first predicted and identified attacks and then determined the most appropriate defense measures for reducing the impact of attacks. The authors of [24] reported the random operation semantics of ADTrees based on stochastic Petri net and performed quantitative analysis. The authors of [25] completed a quantitative analysis of ADTree based on stochastic timed automata. The authors of [26] analyzed the optimal strategies for attackers and defenders in an ADTree based on the game theory model. The authors of [27, 28] performed a quantitative analysis of several attack defense scenarios based on the game theory model. The authors of [29] converted the ADTree into an extended asynchronous multiagent system (EAMAS) and, through this conversion, quantified the impact of different agent configurations on metrics. This type of research must first transform the ADTree into a formal model and then perform quantitative analysis based on the formal model. Due to the complexity of formal models and the state space explosion problem, such methods are difficult to apply in practical cases.

In addition, by utilizing the characteristics of the tree structure, studies have adopted algebraic analysis methods for calculating ADTree's quantitative properties. For example, the authors of [30] applied the ADTree to accurately calculate the damage cost of multistep attacks, measure the propagation of attack damage in the network, and choose appropriate countermeasures for minimizing the impact of attacks on services. For the multiparameter optimization of ADTrees, the authors of [31] designed automation techniques for optimizing all parameters. In addition, the open-source tool ADTool developed in [32] allows users to build various attack scenarios and calculate multiple attributes such as attack time, cost, and probability through recursive algorithms. This open-source tool provides guidance for the defense of attack paths. The existing analysis methods of ADTree are summarized in Table 1.

Current studies based on algebraic methods mainly use the number of defense measures with other economic parameters as factors for evaluating the pros and cons of various defense strategies. However, these works do not consider the selection of a defense strategy from the perspective of the minimum defense cost. In this study, we propose a method for calculating the minimum defense cost of ADTrees based on the algebraic method and implement a calculation tool. This paper is an extended version of [33]. Based on [33], we refined the algorithm, proved the equivalence of A2DTree and ADTree in terms of the minimum defense cost, and completed additional case studies and a comparison to related work.

TABLE 1: Analysis methods of ADTree.

ADTree analysis methods	Theory model	References
Formal methods	Continuous-time Markov chain	[23]
	Stochastic Petri net	[24]
	Stochastic timed automata	[25]
	Game theory model	[26–28]
	An extended asynchronous multiagent system	[29]
Algebraic analysis methods	Binary decision diagram	[30–32]

3. Method Overview

Figure 1 shows the calculation procedure of the minimum defense cost. To calculate the minimum defense cost of a CPS, the proposed A2DTree is used to model attack and defense events in the CPS. Compared to the conventional ADTree, the best characteristic of A2DTree is that only atom attack nodes have defense strategies (also called defense nodes) and all defense nodes are represented as leaf nodes, which makes them efficient for obtaining all potential practical defense devices. In the modeling process, we assume that all attack nodes, defense strategies, and defense costs have been provided by security experts. Our open-source tool can import the provided information as parameters and display the modeling results in a graphical manner for convenient use. Since ADTree is usually used to model attack defense scenarios in industry, we also provide an algorithm for equivalent conversion from ADTree models to A2DTree models. The conversion algorithm and the proof of the conversion equivalence are detailed in Sections 4.2 and 4.3.

Once the modeling is completed, the path sets are to be identified. Similar to other tree models, a path set of the A2DTree is a set of atom attacks nodes, and unless all these nodes fail, the top attack event will not occur. Thus, if all atom attacks in the path set are defended, then the attacks against CPS will fail. Therefore, the final defense strategy of CPS is heavily dependent on the identified path sets. In this study, we efficiently and effectively calculate the path sets via algebraic analysis. Based on the identified path sets, we cumulatively add the defense costs, and the minimum defense cost can be determined. The minimum defense cost calculation algorithm is specified in Section 5.

4. Atom Attack Defense Tree

To calculate the minimum defense cost for CPS, one can use conventional ADTree to establish the system's attack-defense model. After that, a recursive traversal algorithm [32] can be designed to identify the optimal defense nodes that can prevent the top event of the tree model from occurring, and the corresponding cost is minimized. However, this type of solution method requires all subtrees of the ADTree to be queried, which is highly complex and inefficient, as validated by our experiments in Section 6.3. To address this issue, we propose an ADTree structure called A2DTree. Based on this

FIGURE 1: The procedure for calculating the minimum defense cost.

new modeling technology, the minimum defense cost of the system can be easily calculated via algebraic methods. In this section, we first formally define A2DTree. Then, we provide an algorithm for equivalent conversion of ADTree models to A2DTree models. The proof of the conversion equivalence is in section C.

4.1. Definition of A2DTree. A2DTree is a special type of ADTree. It restricts the general ADTree as follows: ① the type of the root node is an attack node; ② only the atom attack node has a corresponding defense node. Figure 2 shows an example of an A2DTree. In the figure, circles with labels A_i are atom attack nodes, and squares with labels D_i are the corresponding defense nodes. The top event of the tree represents the final goal of an attacker (e.g., gaining root access to the system). The circles with labels M_i are intermediate nodes that represent attack results or attacks without defense strategies.

The formal description of the A2DTree is given as follows:

Given an A2DTree, $A2DTree = \langle V, E, Q \rangle$, $V = V^{at} \cup V^{df}$, where V^{at} is the set of attack nodes, and V^{df} is the set of defense nodes. Moreover, $e = \{v_s, v_d\} \in E$, where e is the edge from node v_s to v_d , and $Q = \{AND, OR, SAND\}$,

FIGURE 2: An example of atom attack defense tree.

with Q being the operator function of the ADTree. In addition, $v^r \in V^{at}$, where v^r is the root node, and V^l is the set of atom attack nodes. There exists a one-to-one mapping relationship between V^{df} and V^l .

Cut and path sets provide important information about the vulnerability of the system. The definitions of cut and path sets for the A2DTree are given in the following.

The cut set of an A2DTree is the set of atom attacks that can make the top attack successful.

Definition 1. A cut set of the A2DTree is a set, where $\exists V^c, \forall v_c \in V^c$ and $v_c \in V^l$, and V^l is the set of atom attack events in the A2DTree. If the atom attacks in the set V^c are all successful, then the top attack event will succeed.

The path set of the A2DTree is a set of atom attacks that ensure that the top attack event fails.

Definition 2. A path set of the A2DTree is a set, where $\exists V^p, \forall v_p \in V^p$, and $v_p \in V^l$, and V^l is the set of atom attacks in the A2DTree. If all the atom attacks in set V^p fail, then the top attack goal will fail.

Compared to ADTree, the structure of A2DTree is clearer. The root node of A2DTree represents the goal of the attacker; thus, the meanings of the model are apparent. Particularly, in A2DTree, only atom attack nodes have corresponding defense nodes, and the intermediate nodes are not allowed to have defense nodes. Therefore, we can identify the path set and the minimum defense cost of A2DTree via algebraic methods, which can accelerate the calculation efficiency significantly.

4.2. Conversion of ADTree to A2DTree. Since ADTree is commonly used in industry, a new tree modeling technology may not be easily accepted and used. To help security experts quickly adapt to A2DTree, in this section, we propose an algorithm for converting existing ADTree models into equivalent A2DTree models. As described in the above section, the intermediate nodes are not allowed to have

defense nodes in A2DTree. Hence, to convert the ADTree into an A2DTree, all intermediate attack nodes with defensive child nodes need to be moved down to become leaf nodes (the atom attack node and its corresponding defense node can be considered as a whole as a leaf node). For an intermediate attack node with a defense node, the downward movement process can be divided into 5 steps:

- (1) Construct two intermediate substitute nodes $T1$ and $T2$
- (2) Add the intermediate node $N1$ that needs to be moved down to the child node set of $T1$
- (3) Add the original child nodes of $N1$ to the child node set of $T2$, and the logical relationship between the original child nodes of $N1$ remains unchanged
- (4) Add the $T2$ node to the child node set of $T1$, and the logical relationship between the child nodes of $T1$ is AND
- (5) Add the $T1$ node to the child node set of the original $N1$ parent node

To obtain an A2DTree model of an ADTree model, a recursive traverse is started from the root node of the ADTree, and the abovementioned downward process on all the intermediate nodes with defensive child nodes is performed. Figure 3 shows an example of the conversion of an ADTree model into an A2DTree model.

Algorithm 1 describes the complete conversion process. Suppose that the ADTree to be solved has A attack nodes and D defense nodes. The transformation process actually traverses the entire ADTree and moves the intermediate attack nodes with defense nodes down; thus, the conversion algorithm has a linear time complexity $O(A + D)$.

4.3. Proof of Equivalence. The following proves that after the intermediate nodes are moved down, the logical relationship and the minimum defense cost of the ADTree do not change.

Assume that there is an intermediate attack node $A1$ in the ADTree named ADT; $A1$ has a defense child node and $T1$ is a subtree with the $A1$ node as its root node. Suppose that $T1$ has several subtrees $t1, t2, \dots, tn$, and the logical relationship between the subtrees is $T1 = t1 \wedge t2 \wedge \dots \wedge tn$ or $T1 = t1 \vee t2 \vee \dots \vee tn$. After the attack nodes are moved down, the $T1$ subtree is converted into the $T2$ subtree, $T2 = T1 \wedge TEMP$, and the TEMP subtree is the same as the original $T1$ subtree, $TEMP = t1 \wedge t2 \wedge \dots \wedge tn$ or $TEMP = t1 \vee t2 \vee \dots \vee tn$. Regarding the logical relationship, $TEMP = T1$ and $T2 = T1 \wedge TEMP$; that is, after the $A1$ node moves down, the logical relationship between the original nodes of the ADTree has not changed, and the attack path is the same as that of the original ADTree.

It can be proven that the minimum defense cost of any subtree in the ADTree is equal to the minimum defense cost of the subtree obtained after transformation. Suppose that ADT_1 is a subtree in the ADTree with N_1 as its root node, and N_1 is an intermediate attack node with D_1 as its defense node. ADT_1 can successfully defend at the lowest possible defense cost in two ways: (1) using the D_1 defense node and

FIGURE 3: An example of conversion of an attack defense tree into an atom attack defense tree.

(2) not using the D_1 defense node but using all other combinations of the ADT_1 defense nodes that can successfully defend at the lowest defense cost. The minimum defense costs corresponding to these two schemes are C_1 and C_2 , respectively, and the minimum defense cost $MinCost_1$ of ADT_1 is equal to the lowest cost between C_1 and C_2 . After conversion, ADT_1 becomes ADT_2, and N_1 is moved down to become a leaf node. Suppose that T_1 is the root node of ADT_2 and T_2 is the new parent node of the original child node of N_1 . Because the logical relationship between N_1 and T_2 is AND and T_1 and T_2 have no defensive child nodes, there are two options ADT_2 can use for a successful defense, and the defense cost may be the lowest when (1) using D_1 defense measures or (2) using a combination of defense nodes that can successfully defend at the lowest defense cost in T_2 . The minimum defense costs corresponding to the above two schemes are C_3 and C_4 , respectively, and the minimum defense cost $MinCost_2$ of ADT_2 is equal to the lowest cost between C_3 and C_4 . Because C_1 is equal to C_2 and C_3 is equal to C_4 , $MinCost_1$ and $MinCost_2$ are equal; that is, the minimum defense cost of ADT_1 is the same as the minimum defense cost of ADT_2. Generalizing the above conclusions, we can prove that the minimum defense cost of an ADTree is equal to the minimum defense cost of the corresponding A2DTree.

5. Minimum Defense Cost Calculation

After converting the ADTree into an A2DTree, we can use the success tree method [34] to determine the path set of the A2DTree. We can first identify the dual tree [35] of an A2DTree by replacing all the AND logic gates in the original A2DTree with OR logic gates and replacing all the OR logic gates with AND logic gates. On this basis, the cut set of this dual tree is the path set of the original A2DTree. This study uses the algebraic method to determine the cut set of the ADTree.

The specific steps are as follows:

- (1) By treating the A2DTree attack nodes as Boolean variables, we recursively descend from the root node layer by layer and establish a Boolean expression that represents the root node by an atom attack node
- (2) By expanding the Boolean expression of the root node, we can obtain a disjunctive normal form (DNF)

```

Input ADTree
Output A2DTree
(1) procedure ConverseToA2DTree (Node root)
(2)   Children:= {children of root node};
(3)   NewChildren = {};
(4)   i:= 1;
(5)   repeat
(6)     child:= i-th node in Children;
(7)     if child is an intermediate attack node and
(8)       has defensive child nodes
(9)     then
(10)      T1:= new attack node;
(11)      Child node relationship of T1:= Child
(12)      node relationship of child;
(13)      T1's children:= child's children;
(14)      T2:= new attack node;
(15)      T2 children's operator:= AND;
(16)      Add child to T2's children set;
(17)      Add T1 to T2's children set;
(18)      call conversToA2DTree (T1);
(19)      Add T2 to Newchildren;
(20)    else
(21)      call ConverseToA2DTree (child);
(22)      Add child to Newchildren;
(23)    end if
(24)    i:= i+1;
(25)  until i= the subnode number of Children + 1;
(26)  Collection of child nodes:=NewChildren;
(27) end procedure
(28) NewRoot:= new root node;
(29) Add the ADTree root node to the NewRoot child;
(30) node collection;
(31) call ConverseToA2DTree (NewRoot);

```

ALGORITHM 1: Conversion of an ADTree into an A2DTree.

- (3) The attack nodes corresponding to all the variables of the conjunctive normal form in this DNF constitute a cut set of the A2DTree

After all the path sets of the A2DTree are obtained, the sum of the defense costs corresponding to all the atom attacks in each path set is calculated. The minimum value of all defense costs is the minimum defense cost. The complete process is shown in Algorithm 2.

Assume that the ADTree to be solved has A attack nodes and D defense nodes, among which there are I intermediate attack nodes with defense nodes. The A2DTree converted from the ADTree has $A + 2I$ attack nodes and D defense nodes. The algorithm needs to traverse the entire A2DTree to establish the Boolean expression composed of atom attack nodes to calculate the minimum defense cost; thus, the time complexity of the algorithm is $O(A + 2I + D)$.

6. Case Study and Performance Comparison

The following section shows the execution process of the proposed method through two typical examples and illustrates the effectiveness and efficiency of the method.

6.1. Case Study 1. The following case study considers the bank account example in [12] to verify the method proposed in this paper. Banks aim to protect the accounts of their customers from theft. There are two forms of attacks on personal bank accounts. Attackers can steal funds from accounts through online attacks or ATMs. To steal money through an ATM, the attacker needs a password and a bank card. When customers lose their bank card, they can reduce the loss by reporting the loss of their card. We ignore how attackers obtain bank cards and focus on passwords. When a customer types a password, criminals can steal the customer's password by installing a camera or a special device on the ATM. Regarding this device, the bank can inspect its ATM machines regularly to eliminate the hidden dangers of password theft. Alternatively, an attacker may obtain the note containing the customer's password, and a simple defense measure for preventing passwords from being exposed through notes, for example, is to remember personal passwords.

For ATM theft, the bank can directly add face recognition authentication to its ATMs, and the cardholder himself is required to withdraw money from the ATM. For online banking attacks, the attacker needs to obtain the

```

Input A2DTree
Output The minimum defense cost of the A2DTree and the set of attack nodes that need to be defended
(1) BooleanExpression := A2DTree logical expression;
(2) PathSets := {All simple conjunctions in BooleanExpression, that is, the set of all path sets of the A2DTree};
(3) MinCost :=  $+\infty$ ;
(4) PathSet := {};
(5)  $j := 1$ ;
(6) repeat
(7)   pathset :=  $j$ th path set in PathSets;
(8)   cur_cost := Cut-set defense cost;
(9)   if cur_cost < MinCost
(10)    then
(11)      MinCost := cur_cost;
(12)      PathSet := pathset;
(13)    end if
(14)     $j := j + 1$ ;
(15)  until  $j = \text{Number of subsets in PathSets} + 1$ ;

```

ALGORITHM 2: Calculation of the minimum defense cost of ADTrees.

customer's online banking credentials, including the username (certificate number) and password. Although usernames can be easily retrieved, obtaining the password requires a phishing e-mail or keylogger. For phishing e-mails, certain filtering measures can be adopted on the mail server side. Regarding keyloggers, professional antivirus software can be installed for defense purposes. To prevent password loss or theft, banks can introduce secondary authentication, such as a key fob, for two-factor authentication. A key fob is a small, secure terminal with a built-in authentication mechanism. Its preshared key is known only to the key fob and the bank. Figure 4 shows the ADTree for this example, and Table 2 provides the meanings of all nodes in Figure 4.

Through comprehensive assessment of factors such as the difficulty of the implementation of defense measures and the time and funds required, evaluators rated the defense cost levels according to the actual situation and calculated the defense cost level of each defense node, as summarized in Table 3.

ADTree modeling software is used to model the ADTree. After modeling is completed, the model is exported as an XML file, the ADTree minimum defense cost calculation tool is enabled, and the XML file is imported. Figure 5 shows the file import result interface.

The "Add Defense Cost Attribute" option is selected to add the defense cost value to the defense node. The defense cost value is a nonnegative real number. The user can choose a specific value or the defense cost level according to the demands of the user. Figure 6 shows the attribute assignment result.

After attribute assignment, the "Calculate Defense Cost" option is chosen to calculate the minimum defense cost. All the calculation results will be displayed in the pop-up text box in ascending order of the defense cost. Figure 7 shows select calculation results.

It is evident from the output of the minimum defense cost calculation tool that the node set $\{A1, M4\}$ has the

FIGURE 4: Example of an ADTree: an attack on a bank account.

lowest defense cost. According to the results, the corresponding attack nodes $A1$ and $M4$ in $\{A1, M4\}$ need to be strengthened. As long as nodes $A1$ and $M4$ are successfully defended, the attack target will not be achieved, and this defense strategy attains the lowest cost.

6.2. Case Study 2. The following section adopts the SCADA system of the power system in [36] as a case study to verify the proposed method in this paper. The SCADA system is composed of network components such as the control center network, the communication network between the control center and substation, and the substation automation system. Attackers can take advantage of network component vulnerabilities to attack the SCADA system and obtain illegal operation rights, which could potentially cause power system safety hazards and economic losses [36]. In this example, the attacker issued a trip

TABLE 2: The meaning of each node in Figure 4.

Node label	Meaning
UG	Bank account
M1	ATM
M2	Online
M3	PIN
M4	Password
A1	Card
A2	Username
A3	Eavesdropping
A4	Find note
A5	Phishing
A6	Keylogger
D1	Face recognition
D2	Report loss
D3	Two-factor authentication
D4	Periodic inspection
D5	Memorization
D6	Server-side filtering
D7	Antivirus software

TABLE 3: Defense cost level of Figure 4.

Defense node	Defense cost level
D1	2
D2	1
D3	1
D4	2
D5	1
D6	2
D7	2

FIGURE 5: XML file import results for case study 1.

FIGURE 6: Property assignment result for case study 1.

command to the control protection relay through a network attack, causing the circuit breaker to trip without failure and resulting in a power outage. Figure 8 shows the

Defense-Cost	
{a1 , m4} :	2. 0
{m1 , m4} :	3. 0
{m4 , a2 , a4} :	4. 0
{a1 , a5 , a6} :	5. 0
{m1 , a5 , a6} :	6. 0
{a2 , a4 , a5 , a6} :	7. 0

FIGURE 7: Partial calculation results for case study 1.

FIGURE 8: A2DTree for the SCADA case study.

ADTree obtained by adding defense nodes to the attack tree in [36].

The specific meaning of each node in Figure 8 is shown in Table 4.

Through comprehensive assessment of factors such as the difficulty of the implementation of defense measures and the time and funds required, evaluators rated the defense cost levels according to the actual situation and obtained the defense cost level of each defense node, as listed in Table 5.

ADTree modeling software is used to model the ADTree. After modeling is completed, the model is exported as an XML file, the ADTree minimum defense cost calculation tool is enabled, and the XML file is imported. Figure 9 shows the file import result interface.

The “Add Defense Cost Attribute” option is selected to add the defense cost value to the defense node. The defense cost value is a nonnegative real number. The user can choose a specific value or the defense cost level according to the demands of the user. Figure 10 shows the attribute assignment result.

After attribute assignment, the “Calculate Defense Cost” option is chosen to calculate the minimum defense cost. All the calculation results will be displayed in the pop-up text box in ascending order of the defense cost. Figure 11 shows select calculation results.

It is clear from the output of the minimum defense cost calculation tool that there are two sets of nodes, i.e., {A3, A4, A6, A9, M1} and {A3, A4, A7, A9, M1}, corresponding to the lowest defense cost. According to the results, as long as nodes A3, A4, A6, A9, and M1 are successfully defended or A3, A4, A6, A7, and M1 are all successfully defended, the attack target will not be achieved. The defense costs of these two defense strategies are the same and minimized.

TABLE 4: The meaning of each node in the attack defense tree of Figure 8.

Node label	Meaning
UG	The circuit breaker trips without a fault, resulting in a power outage.
M1	A trip command is sent through the front-end processor.
M2	The status evaluation module is affected, and the operator sends a trip command error.
M3	The human-machine interface (HMI) substation is accessed, and a trip command is sent to the relay.
M4	The remote terminal unit (RTU) is accessed; the relays for RTU monitoring are controlled or the relays are reconfigured.
M5	Direct access to the relay protector is obtained.
M6	False data are injected.
A1	The hardware firewall is bypassed for port scanning.
A2	The control center application server is accessed.
A3	Measurement and status packets are intercepted.
A4	An eavesdropping device is installed.
A5	The encrypted message is decoded.
A6	Port scanning is implemented.
A7	The substation user interface is accessed.
A8	A connection via dial-up is established.
A9	The password is decoded.
A10	Port scanning is conducted.
A11	The password is decoded.
D1	Idle and potentially threatening ports are disabled, and the firewall is used to mask scanned packets.
D2	Server data are backed up and server security measures are enhanced.
D3	Measurements are conducted and packet encryption measures are implemented.
D4	An antieavesdropping cable, an encryption algorithm, or an antieavesdropping device is implemented.
D5	A better encryption algorithm is adopted.
D6	Idle and potentially threatening ports are disabled, and the firewall masks scanned packets.
D7	The router is enhanced to prevent IP scanning.
D8	Strong modem encryption is adopted.
D9	A new encryption algorithm is applied, including RTU mandatory authentication.
D10	The protection of relay authorized access is realized.
D11	Strong passwords are selected for the network.
D12	Advanced permissions are included for trip commands.
D13	Data digital signature protocols are established.
D14	Scanning is conducted to fix any vulnerabilities in the HMI.
D15	The RTU firmware is updated and a security gateway is deployed.
D16	The relay protector firmware is updated on time.

TABLE 5: Defense cost level of the defense nodes in Figure 8.

Defense node	Defense cost level
D1	2
D2	3
D3	1
D4	4
D5	2
D6	2
D7	3
D8	1
D9	1
D10	4
D11	1
D12	1
D13	2
D14	3
D15	4
D16	4

6.3. Performance Comparison. We first give the time and space complexity of our method. Suppose that the ADTree to be analyzed has A attack nodes and D defense nodes, among which there are I intermediate attack nodes with defense

FIGURE 9: XML file import results for case study 2.

FIGURE 10: Property assignment result for case study 2.

```

{a3 , a4 , a6 , a9 , m1} : 6.0
{a3 , a4 , a7 , a9 , m1} : 6.0
{a3 , a5 , a7 , a9 , m1} : 7.0
{a1 , a3 , a4 , a6 , a9} : 7.0
{a3 , a7 , a9 , m1 , m3} : 7.0
{a3 , a6 , a9 , m1 , m3} : 7.0
{a1 , a3 , a4 , a7 , a9} : 7.0
{a4 , a7 , a9 , m1 , m6} : 7.0
{a4 , a6 , a9 , m1 , m6} : 7.0
{a3 , a5 , a6 , a9 , m1} : 7.0

```

FIGURE 11: Partial calculation results for case study 2.

TABLE 6: Time and space consumption of the two algorithms.

ADTree	Number of attack nodes	Number of defense nodes	Number of logic gates	Time (ms)		Space (k)	
				CONV_ALGO	UTDRE_ALGO	CONV_ALGO	UTDRE_ALGO
A	89	67	44	635	8733	61828	119028
B	48	41	19	563	2568	79120	136852
C	64	57	25	4731	25483	170016	129104
D	51	39	22	1021	24172	74912	162644
E	61	53	24	16119	44439	155124	156796

nodes. As noted in Section 6.2 and Section 5, Algorithm 1 has a linear time complexity $O(A + D)$, and the time complexity of Algorithm 2 is $O(A + 2I + D)$. Further, suppose that the analyzed ADTree has N path sets, and each path set has M elements, the time complexity of defense cost computation is $O(M * N)$. Therefore, the overall time complexity of our proposed method is $O(2A + 2I + 2D + M * N)$. The space complexity of Algorithm 1 is $O(2I)$, and the complexity of Algorithm 2 is $O(1)$. Hence, the space complexity of the proposed method is $O(2I + 1)$.

Then, we compare our method with ADTool [32], which is a popular open-source tool for attack tree analysis. ADTool uses a top-down recursive algorithm, i.e., UTDRE_ALGO, which calculates the path set of all the subtrees containing the original tree root nodes. For comparison, we enhanced ADTool by adding a function for calculating the sum of the defense costs in each path set and obtaining the minimum defense cost. This function is invoked at the last of UTDRE_ALGO. In our proposed method, we use two algorithms (Algorithms 1 and 2) to calculate the minimum defense cost of an ADTree. For simplicity, we will call our algorithms as CONV_ALGO. To assess the pros and cons of the two algorithms (CONV_ALGO and UTDRE_ALGO), we use the two algorithms to determine the minimum defense cost of five ADTree models. The specific information on the models is provided in Table 6, and the algorithm time and space efficiency are calculated. All the experiments in this paper were performed on a computer with four cores and sixteen threads, a CPU frequency of 2.6 GHz, and a memory of 16 Gb. The experimental results are summarized in Table 6 and shown in Figure 12.

According to the experimental results, the time consumption of CONV_ALGO is better than that of UTDRE_ALGO. From the perspective of time complexity analysis, the time complexity of UTDRE_ALGO is related to

FIGURE 12: Comparison of the time and space consumption of our method and ADTool.

the number of subtrees containing root nodes in the ADTree, and the time complexity of CONV_ALGO is related to the number of defense nodes in the ADTree. When the size of the ADTree is large, the number of subtrees will be large. UTDRE_ALGO must calculate the path set of numerous sub-ADTrees, while CONV_ALGO needs to calculate only the path set of the transformed A2DTree, thereby reducing the time required to calculate the path set and improving the efficiency.

7. Discussion

In our study, we mainly use the ADTree as an attack modeling tool, which can be accomplished because ADTree can model system attack-defense scenarios, and they have been widely used in the industry [37, 38]. For the other models, such as the attack graph [39], one can first transform it into a tree model and then apply our method to calculate the minimum defense cost. The transformation is straightforward, and the commonalities between the attack tree and attack graph are illustrated in [40, 41].

In this paper, we directly apply the defense cost, and we do not specifically consider how the defense cost is obtained. One can refer to [42, 43] for details on the defense cost calculation. In practice, the defense cost refers to the actual cost of the defender in a complete attack defense scenario. Hence, it can be a specific value, such as the hardware equipment, software development, labor, and time costs. The defense cost can also be a relative value by considering only the cost level of each cost item. Both types of defense costs are supported in our method.

Our method also has certain limitations, which we hope to address in future work. For example, our method considers only the minimum defense cost and identifies the optimal defense nodes. In real-world applications, one should consider adding redundant defense measures to ensure the robustness of the security protection system. Therefore, other factors, such as robustness, should be considered during the optimization process.

8. Conclusion

This paper focuses on the assessment of the CPS security defense cost and combines ADTree modeling and path set calculation approaches to establish a minimum defense cost calculation method suitable for CPSs. First, based on ADTree, A2DTree is proposed by moving all the intermediate nodes with defense nodes down to the leaf nodes, and the equivalence of the transformation is proved. On this basis, a minimum defense cost calculation algorithm is provided, and an open-source calculation tool is implemented. Finally, the effectiveness of our method is illustrated in two typical examples, and the efficiency of our method is demonstrated by experimental comparison with related work. The main tasks in the future are the improvement of the A2DTree structure and consideration of the minimum A2DTree defense cost with sequential logic.

Data Availability

The data in case study 1 and 2 of this article are from [9, 33], respectively. They are publicly available. The tool we developed has been open-sourced in GitHub, and the download URL is https://github.com/zzc1/ADTree_Min_DefCost/releases/tag/1.0.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 61802192 and 61702282), by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant no. 18KJB520024), by Nanjing Forestry University (Grant nos. GXL016 and CX2016026), by NUPTSF (Grant no. NY217143), and by Undergraduate Innovation Training Program of NJFU (Grant no. 2019NFUSPITP0581).

References

- [1] R. Alguliyev, Y. Imamverdiyev, and L. Sukhostat, "Cyber-physical systems and their security issues," *Computers in Industry*, vol. 100, pp. 212–223, 2018.
- [2] R. Pal and V. Prasanna, "The STREAM mechanism for CPS security the case of the smart grid," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 4, pp. 537–550, 2016.
- [3] M. S. Mahmoud, M. M. Hamdan, and U. A. Baroudi, "Modeling and control of cyber-physical systems subject to cyber attacks: A survey of recent advances and challenges," *Neurocomputing*, vol. 338, pp. 101–115, 2019.
- [4] M. Numan, F. Subhan, W. Z. Khan et al., "A systematic review on clone node detection in static wireless sensor networks," *IEEE Access*, vol. 8, pp. 65450–65461, 2020.
- [5] S. Bhattacharya, R. Kaluri, S. Singh et al., "A Novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU," *Electronics*, vol. 9, no. 2, pp. 1–16, 2020.
- [6] C. Iwendi, Z. Jalil, A. R. Javed et al., "KeySplitWatermark: Zero watermarking algorithm for software protection against cyber-attacks," *IEEE Access*, vol. 8, pp. 72650–72660, 2020.
- [7] V. Bolbot, G. Theotokatos, L. M. Bujorianu et al., "Vulnerabilities and safety assurance methods in cyber-physical systems: A comprehensive review," *Reliability Engineering & System Safety*, vol. 182, pp. 179–193, 2018.
- [8] T. Liu, J. Tian, J. Z. Wang et al., "Integrated security threats and defense of cyber-physical systems," *Acta Automatica Sinica*, vol. 45, no. 1, pp. 5–24, 2019.
- [9] Q. L. Guo, S. J. Xin, J. H. Wang et al., "Comprehensive security assessment for a cyber physical energy system: A lesson from Ukraine's blackout," *Automation of Electric Power Systems*, vol. 40, no. 5, p. 145G14, 2016.
- [10] M. R. Sheibani, G. R. Yousefi, M. A. Latify, and S. Hacopian Dolatabadi, "Energy storage system expansion planning in power systems: A review," *IET Renewable Power Generation*, vol. 12, no. 11, pp. 1203–1221, 2018.
- [11] T. Lins and R. A. R. Oliveira, "Cyber-physical production systems retrofitting in context of industry 4.0," *Computers & Industrial Engineering*, vol. 139, Article ID 106193, 2020.
- [12] B. Kordy, S. Mauw, S. Radomirovic, and P. Schweitzer, "Attack-defense trees," *Journal of Logic and Computation*, vol. 24, no. 1, pp. 55–87, 2014.
- [13] A. Shameli-Sendi, H. Louafi, W. He, and M. Cheriet, "Dynamic optimal countermeasure selection for intrusion response system," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 755–770, 2018.
- [14] P. Wang and J. C. Liu, "Improvements of attack-defense trees for threat analysis," *Advances in Intelligent Systems and Applications-Volume 2*, Springer, Heidelberg, Berlin, Germany, pp. 91–100, 2013.

- [15] N. Chakraborty and E. Kalaimannan, "Minimum cost security measurements for attack tree based threat models in smart grid," in *Proceedings of the IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, pp. 614–618, New York, NY, USA, October 2017.
- [16] P. Wang, W. H. Lin, P. T. Kuo et al., "Threat risk analysis for cloud security based on attack-defense trees," in *Proceedings of the 8th International Conference on Computing Technology and Information Management (NCM and ICNIT)*, vol. 1, pp. 106–111, Seoul, Republic of Korea, April 2012.
- [17] H.-K. Kong, M. K. Hong, and T.-S. Kim, "Security risk assessment framework for smart car using the attack tree analysis," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 3, pp. 531–551, 2018.
- [18] S. Wang, Z. Zhang, and Y. Kadobayashi, "Exploring attack graph for cost-benefit security hardening: a probabilistic approach," *Computers & Security*, vol. 32, pp. 158–169, 2013.
- [19] B. Schneier, "Attack trees," *Dr. Dobbs's Journal*, vol. 24, no. 12, pp. 21–29, 1999.
- [20] S. Mauw and M. Oostdijk, "Foundations of attack trees," *Information Security and Cryptology - ICISC 2005*, Springer, Heidelberg, Berlin, Germany, pp. 186–198, 2005.
- [21] J. Bryans, H. N. Nguyen, and S. Shaikh, "Attack defense trees with sequential conjunction," in *Proceedings of the IEEE 19th International Symposium on High Assurance Systems Engineering (HASE)*, pp. 247–252, Hangzhou, China., January 2019.
- [22] B. Kordy, S. Mauw, and P. Schweitzer, "Quantitative questions on attack–defense trees," in *Proceedings of the International Conference on Information Security and Cryptology*, pp. 49–64, Seoul, Republic of Korea, December 2012.
- [23] R. Jhawar, K. Lounis, and S. Mauw, "A stochastic framework for quantitative analysis of attack-defense trees," in *Proceedings of the International Workshop on Security and Trust Management*, pp. 138–153, Guildford, England, 2016.
- [24] K. Lounis, "Stochastic-based semantics of attack-defense trees for security assessment," *Electronic Notes in Theoretical Computer Science*, vol. 337, pp. 135–154, 2018.
- [25] R. R. Hansen, P. G. Jensen, K. G. Larsen et al., "Quantitative evaluation of attack defense trees using stochastic timed automata," in *Proceedings of the International Workshop on Graphical Models for Security*, pp. 75–90, Hoboken, NJ, USA, July 2017.
- [26] H. P. Huang, S. D. Xiao, and X. Y. Meng, "Cyber security assessment for SCADA systems based on attack-defense game model," *Computer Engineering & Science*, vol. 39, no. 5, pp. 877–884, 2017.
- [27] Z. Aslanyan, F. Nielson, and D. Parker, "Quantitative verification and synthesis of attack-defence scenarios," in *Proceedings of the IEEE 29th computer security foundations symposium (CSF)*, pp. 105–119, Lisbon, Portugal, June 2016.
- [28] S. Du, X. Li, J. Du, and H. Zhu, "An attack-and-defence game for security assessment in vehicular ad hoc networks," *Peer-to-peer Networking and Applications*, vol. 7, no. 3, pp. 215–228, 2014.
- [29] J. Arias, C. E. Budde, W. Penczek et al., "Hackers vs. security: attack-defence trees as asynchronous multi-agent systems," pp. 1–19, 2019, <http://arxiv.org/abs/1906.05283>.
- [30] A. Shamel-Sendi, H. Louafi, W. He, and M. Cheriet, "A defense-centric model for multi-step attack damage cost evaluation," in *Proceedings of the 3rd International Conference on Future Internet of Things and Cloud*, pp. 145–149, Rome, Italy, August 2015.
- [31] Z. Aslanyan and F. Nielson, "Pareto efficient solutions of attack-defence trees," in *Proceedings of the International Conference on Principles of Security and Trust*, pp. 95–114, London, UK, 2015.
- [32] B. Kordy, P. Kordy, S. Mauw et al., "ADTool: security analysis with attack–defense trees," in *Proceedings of the International Conference on Quantitative Evaluation of Systems*, pp. 173–176, Buenos Aires, Argentina, August 2013.
- [33] B. Xu, Z. Zhong, and G. He, "A minimum defense cost calculation method for cyber physical system," in *Proceedings of the Seventh International Conference on Advanced Cloud and Big Data (CBD)*, pp. 192–197, Suzhou, China, September 2019.
- [34] W. A. Hyman, "Success tree analysis," *Journal of Clinical Engineering*, vol. 41, no. 1, pp. 40–42, 2016.
- [35] J. Xu, M. Ye, X. Peng, and Z. Li, "Influential factor analysis of China's unsustainable electric power system: A case study of Chengdu electric Bureau," *Energy Policy*, vol. 129, pp. 975–984, 2019.
- [36] M. Ding, X. J. Li, and J. J. Zhang, "Effect of SCADA-oriented cyber attack on power system reliability," *Power System Protection and Control*, vol. 46, no. 11, pp. 37–45, 2018.
- [37] M. Fraile, M. Ford, O. Gadyatskaya et al., "Using attack-defense trees to analyze threats and countermeasures in an ATM: A case study," in *Proceedings of the IFIP Working Conference on the Practice of Enterprise Modeling*, pp. 326–334, Skövde, Sweden, November 2016.
- [38] X. Ji, H. Q. Yu, G. S. Fan, and W. H. Fu, "Attack-defense trees based cyber security analysis for CPSs," in *Proceedings of the 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pp. 693–698, Shanghai, China, June 2016.
- [39] A. T. Al Ghazo, M. Ibrahim, H. Ren, and R. Kumar, "A2G2V: automatic attack graph generation and visualization and its applications to computer and SCADA networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–11, 2019.
- [40] M. S. Haque and T. Atkison, "An evolutionary approach of attack graph to attack tree conversion," *International Journal of Computer Network and Information Security*, vol. 9, no. 11, pp. 1–16, 2017.
- [41] H. S. Lallie, K. Debattista, and J. Bal, "A review of attack graph and attack tree visual syntax in cyber security," *Computer Science Review*, vol. 35, Article ID 100219, 2020.
- [42] A. Bagnato, B. Kordy, P. H. Meland, and P. Schweitzer, "Attribute decoration of attack-defense trees," *International Journal of Secure Software Engineering*, vol. 3, no. 2, pp. 1–35, 2012.
- [43] A. Buldas, O. Gadyatskaya, A. Lenin, S. Mauw, and R. Trujillo-Rasua, "Attribute evaluation on attack trees with incomplete information," *Computers & Security*, vol. 88, Article ID 101630, 2020.