Restructuring Attack Trees to Identify Incorrect or Missing Relationships between Nodes

A Thesis Submitted to the Department of Computer Science and Communications Engineering, the Graduate School of Fundamental Science and Engineering of Waseda University in Partial Fulfillment of the Requirements for the Degree of Master of Engineering

Submission Date: February 1st, 2019

CAI Hua
(5117FG10-0)

Advisor: Prof. Yoshiaki Fukazawa

Contents

Contents 2
1. Introduction 2
2. Background..4
2.1 Motivating Example 4
2.2 Attack Tree 5
2.3 Define the Relationships between Nodes 6
2.4 Analyze Attack Tree 7
3. Related Work..10
4. Approach 12
4.1 Interpretive Structural Modeling (ISM) 12
4.2 ISM on Attack Tree 14
4.2.1 Subtree Physical Access 15
4.2.2 Subtree Invade the Computer 18
5. Discussion..21
6. Conclusion 23
Acknowledgements..24
References..25
Publications..26

1. Introduction

Numerous high-tech products and technologies are invented to improve life, including physical software and cyber-physical systems in the information technology field. Reliance on these types of systems has been increasing. Today these systems store vast amounts of information. However, security of these systems has become important and is being actively researched.

The concept of an attack tree provides a clear way to de-scribe possible attacks and safety threats for a system. We strive to apply the attack tree concept to design a safety system. An attack tree has a structure similar to a tree as it includes different levels of nodes – root nodes and leaf nodes. Root nodes represent attack targets, while leaf nodes denote attack methods.

A root indicates a target goal, and the leaves represent basic attacks with no further refinements in an attack tree. A successful attack would satisfy the roots. That is, if the child nodes are true, then the direct parent nodes are true.

According to the attack tree model, each node can have different metrics, and finding the minimum value of the sum of the metrics passing through the path from the tree's bottom to the top can indicate a system's attack motion [1]. In real cases, attack trees have huge and complicated structures. One attack tree may be composed of thousands of nodes. Due to the complexity of systems, even a specialist can make mistakes while building an attack tree or analyzing its structure, resulting in a chaotic incomprehensible attack tree.

Generally, an attack tree connects a child node to its parent node. Thus, a child and parent have a vertical relationship. However, some relations exist between horizontal nodes. If the horizontal relationships between nodes are considered, the relationship definition of the nodes on the attack tree becomes more complicated. Additionally, the efficiency becomes lower or defenses overlap. If we can redefine the horizontal relation-ships on an attack tree and re-build the tree, a clearer attack tree with a strict structure could be derived, especially for complicate cases.

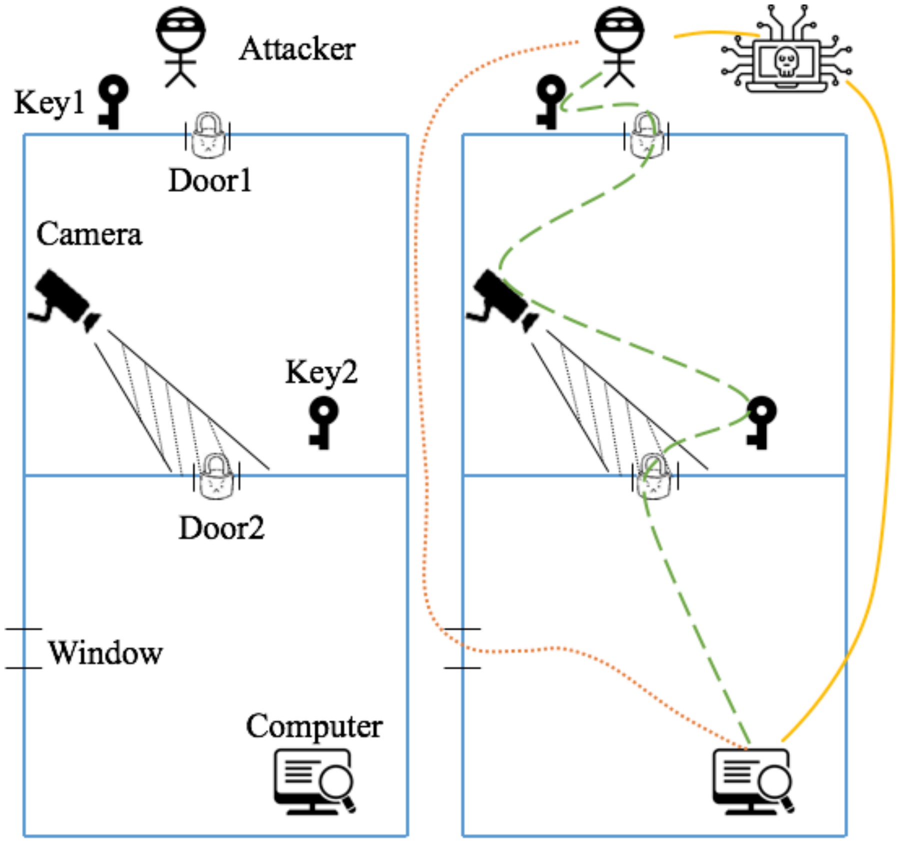
Herein we present a new method to rebuild the initial at-tack tree by the Interpretive Structural Modeling procedure (abbr. ISM). The re-built attack tree can repair incorrect relation between parent-child nodes and it can effectively connect the relationship between nodes. Generating an attack tree with a clear structure can improve the efficiency of data analysis of a system's threat. We show a real example to verify the value of this method.

Section 2 explains the background of an attack tree and the motivating example. Section 3 presents the related work on attack trees. Section 4 how our method works and implements our method using a running example, and Section 5 presents discusses our method. Finally, Section 6 concludes the paper.

2. Background

2.1 Motivating Example

Before explaining our method, we would like to introduce our motivating example. In the running example, there is a data computer in a building. As shown in Fig. 1a, this building includes two rooms (room1 and room2) with two locked doors (door1 and door2). There are two keys (key1 and key2) to open the doors. A camera monitors door2 to make sure no one enters room2. However, the window can be opened freely and is not within the camera's view. The goal for an attacker is to obtain computer data without detection on the camera.



(a)Floor plan (b)Attack scenarios

Fig. 1. Example of a figure caption

Based on these conditions, we can draw three successful scenarios (Fig. 1b). The first is enter room2 through the window (orange dotted line). The second is to use both key1 and key2; the attacker enters room1 using key1 to open door1. Once in room1, the attacker disables the camera and then uses key2 to enter room2 (green dashed line). The third scenario is the attacker hacks directly into the network (yellow solid line). An attack tree can be used to clearly model these three scenarios.

This example is referenced by "Is My Attack Tree Correct?". That paper defines the correctness of an attack tree by nodes relations and confirms it by giving the opposite example, which does not satisfy the necessary and sufficient relationship between nodes. Herein we focus on how to revise an attack tree that does not satisfy the conditions and reduce the amount of detailed descriptions in the theorem proofs. We slightly adjust the model for ease of expression.

2.2 Attack Tree

There are numerous ways to initiate an attack, but most people do not realize this. Even if people are aware of such vulnerabilities, they know little about system security. Due to the increasing complexity of attack motions, it is becoming more difficult to prevent attacks. Defending against an attack based on an engineer's own personal ability is problematic. "Security" must answer two questions: "Who will attack?" and "How long is security necessary?" Thus, a method to model threats against computer systems must be developed. If the attackers are known, then systems could be protected by presuming their ability and their attack target. By knowing the probability of attacking motions, a defense can be mounted by designing these countermeasures. These situations are why the "attack tree" method is needed.

To analyze the security of an issue, the attack tree model must be selected. In our motivating example, we used three types of nodes:

- OR: OR node means an alternative attack. One of the child nodes can be satisfied to attack a parent node.
- AND: AND node means a conjunctive attack. All of the child nodes must be satisfied to attack the parent mode.
- SAND: SAND node means a sequential attack. All of the child nodes must be satisfied in a specific order to attack the parent node.

Fig. 2 expresses the three scenarios in Fig. 1b as an attack tree. To access the computer data, methods can be separated into physical attacks and network attacks. In the physical at-tack case, it can be further separated as go through the door or the window. To use the door, the camera should be disabled before entering room2 (node type AND); to enter room2, door1 and door2 should be opened in sequence (node type SAND). The invade computer subtree can be separated to in-vade from wireless LAN or infection malware. These two nodes can be true if at least one of the leaf nodes is true (node type OR).

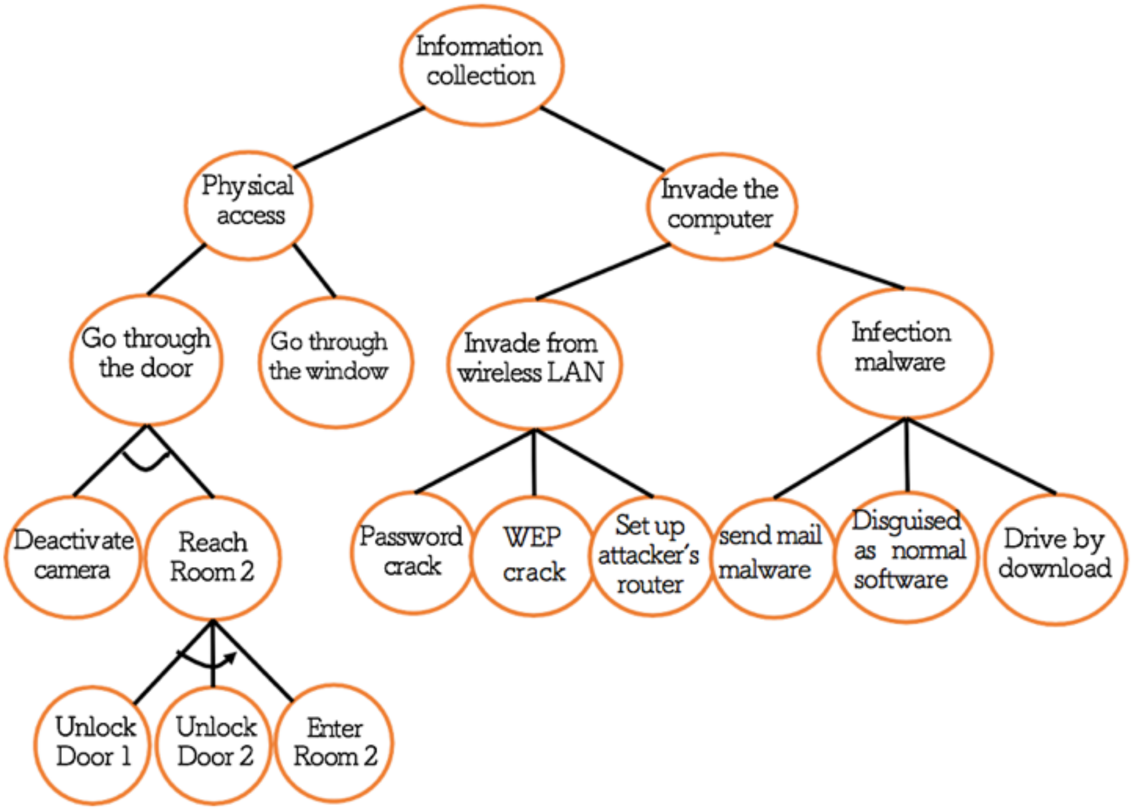


Fig. 2. Attack tree expressed by scenario

2.3 Define the Relationships between Nodes

To discuss the relation of nodes, we define the Necessary and Sufficient Relationship between Parent-Child Nodes and the Horizontal Relationship

1) Necessary and Sufficient Relationship between Parent-Child Nodes: The logic relationship between parent-child nodes is very important in an attack tree. Below are two relationship definitions between parent-child nodes from the attack tree to clarify the child nodes are conditions that must be satisfied to make the direct parent node true.

a) Child nodes' commands must be a sufficient condition for the commands of parent nodes. For an "OR" node, the statement of the parent node is true when at least one of the child nodes is reached. For an "AND" node, the statement of the parent node is true when all child nodes are reached.

b) A set of child nodes must be the necessary condition for the commands of parent nodes. Based on the premise that the attack tree has integrity when the node type is "OR", the un-ion of child nodes is true if the parent node is true, but when the node type is "AND", the intersection of child nodes is true if the parent node is true.

2) Horizontal relationship: The structure of an attack tree is like a tree. The connections between nodes are usually defined as parent-child relationships (vertical relationships). However, some relations exist between nodes, which are at the same level. Such relations, which are usually ignored in the relationship definition, are horizontal relationships between nodes. To redefine the relationships in a tree structure, surplus nodes make it more complicated, reducing the efficiency when set-ting the defense strategy. Horizontal relations are separated into two types as shown below:

a) Additional contributions: One node can be satisfied if another node on a different branch is true.

b) Similar nodes: The same attack occurs from two leaf nodes.

2.4 Analyze Attack Tree

There are multiple ways to analyze attack trees. Schneier [2] provided a method to realize loopholes of the system by estimating the different node variables. In attack trees, every node has its own values, which might be continuous or Boolean. Nodes are decided by different variables such as the lowest qualified attack path for equipment, cheapest attack path, highest success rate attack path, and lowest ability attack path. We can also set different weights for variables to get the lowest integrated cost attack tree by the characteristics of the system. In an attack tree, the root nodes represent the attack target, and the child nodes are an attack or a parent node's sub-goal. A leaf node represents an attack that cannot be disassembled.

When these conditions are combined, *Gn*, which stands for target node, can be defined. *An* stands for leaf node (an attack). *λ(n)* represents the variable cost at point n. *λ* denotes a variable for risk, skill, or other factors. For example, *Risk(Aa)* can be considered as the risk cost of attack node a. For a target node *Gn*, there are *Gx*, *Gy* sub-nodes, and the type of node is OR. The relation function is described as:

$$\lambda(Gn) = Min\{\lambda(Gx), \lambda(Gy)\}$$

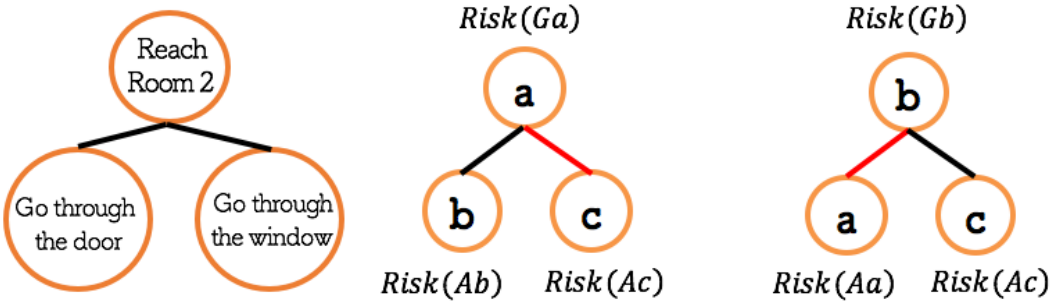
In Fig. 3a, which shows a very simple attack tree with a target of "reach room2", there are two attacks: "go through the door" or "go through the window". We can use the symbols in Fig. 3b to derive the relation function as:

$$Risk(Ga) = Min\{Risk(Ab), Risk(Ac)\}$$

The minimum risk path is chosen between nodes Ab and Ac. When *Risk(Ab)* is bigger than *Risk(Ac),* the minimum risk path is *Path(Ga, Ac).* However, if the parent-child relations are the opposite (Fig. 3c), then the relation function is:

$$Risk(Gb) \ = \ Min\{Risk(Aa), Risk(Ac)\}$$

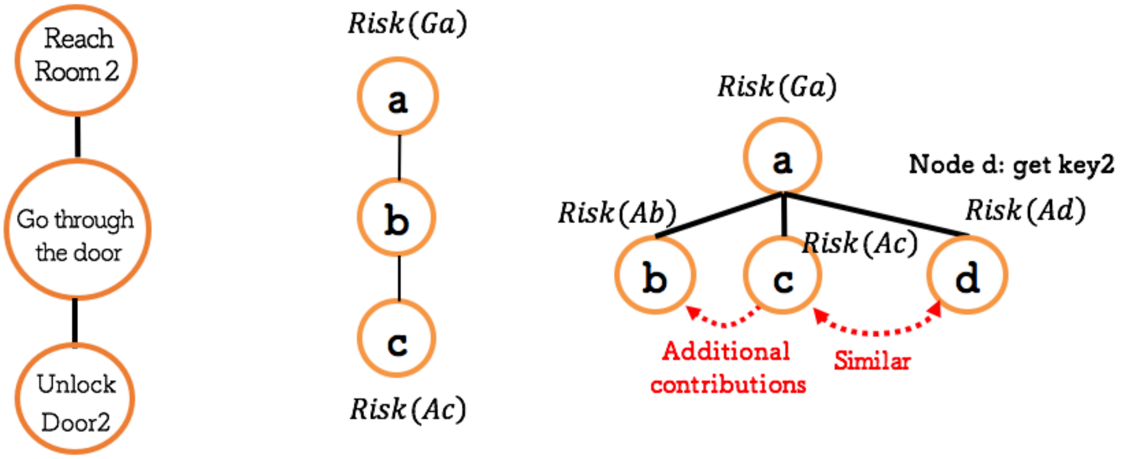
Thus, the minimum risk path can be chosen between nodes *Aa* and *Ac*. When *Risk(Aa)* is greater than *Risk(Ac),* then the minimum risk path is *Path(Gb, Aa).* We found that this path is not the path of "go through the window", which is actually the minimum risk path of the system.



(a) correct attack tree (b) Indicate the nodes (c)parent-child relations opposite

Fig. 3. Example of analyze an incorrect attack tree

In Fig. 4a, there are three nodes from the top to bottom: "reach room2", "go through the door" and "unlock door2", and all of them satisfied the definition of *Necessary and Sufficient Relationship between Parent-Child Nodes*. We changed Fig. 4a to Fig. 4b by symbolization. When we analyze Fig. 4b, we just need to analyze *Risk(Ac)* for getting *Risk(Ga)*. We put figure 4b's node c on the position of node b's siblings purposely, and adding node d: "get key2", which has the same relations as node c. While analyzing the attack tree on Fig. 4c, we need to analyze these three nodes of risk for each three paths. Although it would get the same result probably from *Risk(Ac)* and *Risk(Ad)*, it could cost much more time to analyze the variable value of surplus nodes.



(a) correct attack tree (b) Indicate the nodes (c)with horizontal relation-ship

Fig. 4. Example of analyze attack tree with missing horizontal relationship

By the method from this thesis, we define Necessary and Sufficient Relationship between Parent-Child Nodes and some ignorable relations between horizontal nodes. When analyzers find out these relations, he would re-judge the relationship between nodes and restructure the attack tree.

Consider the physical attack portion of Fig. 2. To satisfy the node "go through the door", two conditions are necessary: "deactivate camera" and "reach room2". However, the condition "reach the room2 " can be achieved when either "go through the door" or "go through the window" is true. Based on the result, it can be said that the condition in this case is a necessary condition but not a sufficient condition. In other words, there are some obvious logical gaps in the attack tree. Thus, the attack tree should be revised.

Consider the invade the computer part portion in Fig. 2. Here "WEP crack" is a child-node belonging to the node "in-vade from wireless LAN". However, it can be viewed as child-node belonging to its siblings-node -- "password crack" be-cause the "password crack" would be true if node "WEP crack" is completed. Ignoring the relationship of additional contributions, makes the attack tree more complicated by increasing of amount of attack paths, reducing the efficiency of setting the defense strategy. In the "Infection malware" subtree, it can be observed that node "disguised as normal software" and node "drive by download" are the same attack on the system. These kinds of designs are messy and waste defense resources by over-comprehending the values of the attack motions. To ad-dress these problems, nodes need to be redefined or combined.

In a real situation, the attack tree would be a huge system with tons of nodes. To prevent the aforementioned problems, the tree's path needs to have a clear structure to elucidate the relations between nodes. In the next section, we offer a new way to re-build an attack tree with incorrect relationships between parent-child nodes and analyze the horizontal relationships of its nodes.

3. Related Work

Earlier works introduced the concept of an attack tree. Salter et al. [3] described how to characterize adversaries and map vulnerabilities of an information system. Schneier [2] presented attack tree modeling and analysis by considering cost; their tree can determine where a system is vulnerable. Jurgenson and Willemson [4] extended the multi-parameter attack tree model by considering the skill level, cost, and feasibility of an attacker to work with interval estimates instead of point estimates. They also discussed how to choose the optimal or an economically justified security level. Jurgenson and Willemson [5] extended the standard attack tree, which can order to the attacker's decision-making process. Based on a sequence attack tree, they proposed an algorithm to compute the attacker's expected outcome. To reduce the visual complexity of an attack graph, Noel et al. [6] described a number of new attack graph visualization techniques, and their method can be applied alone or combined into coordinated attack graph views. Despite being an intuitive aid in threat analysis, attack trees have yet to be provided with unambiguous semantics. Maui and Oostdjk [7] used mapping of attack suites to be provided denotational semantics, which are abstracted from the internal structure of an attack tree to help participants precisely under-stand how attack trees can be manipulated during construction and analysis.

Attack trees can be also used in copyright protection of e-commerce protocols. Higuero M. V. et al. [8][9] provided a model called SecDP to solve the threat of ambiguity and avoid MITM attacks. They are also used in the mobile system fields. Duhoe Kim et al. [10] used an attack tree to build the analysis for a LOG analysis system to avoid being attacked on Android OS smart phones. The attack tree was separated for the pre-phase, which carried out the steps before an attack, and the post-phase, which detected malware. It has also been used in smart cars. Although few hacking events occurring in a smart car system, it should be noted that various security threats are invading electronic systems, especially for automobiles that can be connected into external networks. HeeKyung Kong et al. [11] proposed a method by modeling a hacking attack logically and creating a security risk assessment frame-work to defend, evaluate, and categorize the security threats and vulnerabilities by attack tree analysis.

4. Approach

We present a method that not only solves problems to re-pair the incorrect relations between parent-child nodes but also re-defines horizontal relationships between nodes on different branches. Our method provides a clear, complete structure of the attack tree through Interpretive Structural Modeling (ISM) procedure.

4.1 Interpretive Structural Modeling (ISM)

The first experiment was evaluating the method in terms of reading comprehension and coding skills. Twelve subjects with some experience in programming were invited through Twitter to participate. The subjects consisted of students, programmers, and other people from different jobs. All were familiar with Java, but none had experience with Swift. They were randomly divided into two groups: A and B. Group A did not have translational support, but group B did. They were asked to do self-study using Jasmin, then their knowledge of Swift language was tested in terms of reading comprehension questions and coding tasks.

ISM is a method to confirm complex relations between items by modeling. For complex issues or systems, the number of items is huge and the relations between items are complicated. To express the relations more precisely, we use ISM to present the relations between items directly and indirectly. By considering the analyzing issue and system, items are initially defined as apexes V_i and V_j (i=1, 2, 3 ⋯ ; j=1, 2, 3 ⋯) for ISM. The edge with an arrow (V_i, V_j) expresses the relations between two items. A matrix, which is used to express the connection status for every pair item, is called a Relation Matrix (A) and it provides a structural self-interaction matrix (SSIM). Afterward, a reachability matrix (RX) from SSIM is obtained by calculations. Finally, we can derive a new model from RX by level partitions. ISM is a very efficient method. Because it is based on different situations, the number of relations between items can be reduced by 50–80% [12]. Additionally, it is a useful tool considering and thinking more logically on complex issues and situations.

ISM includes the following steps:

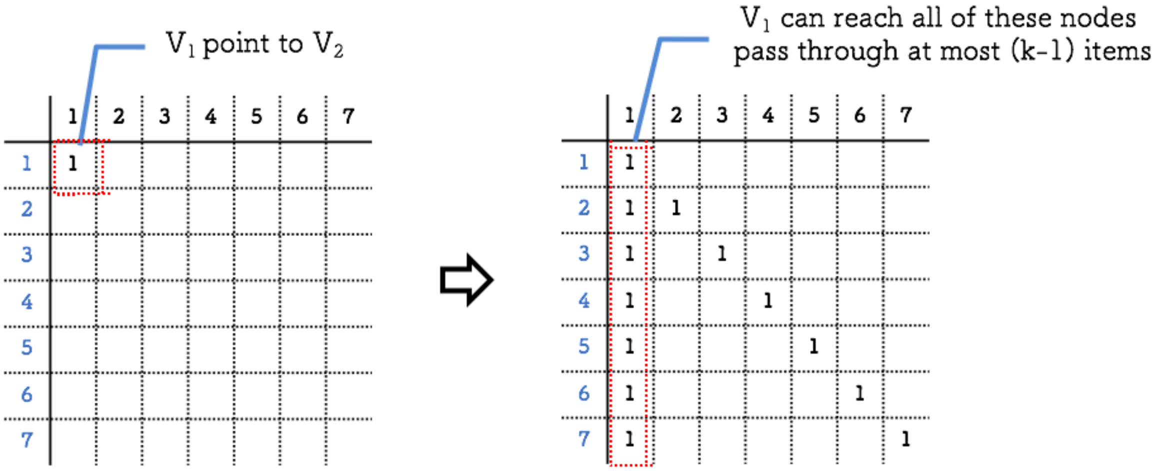
1) Graph the Relation Matrix (A) of the analyzed system

2) Generate the Reachability Matrix (RX) through a trans-formation of the Relation Matrix

3) Level Partitions

4) Digraph

Step 1: Graph the *Relation Matrix (A):* In ISM, a participant or expert familiar with the analysis issue must identify the contextual relations among items. Based on these relations, Relation Matrix A = {a_{ij} | i, j=1,2,･･･, n}is created to express all direct binary relationships. The value of n is the number of nodes from the system. All elements are listed in the rows and columns of a table. If two elements have a direct relation, enter 1 on the table grid. Otherwise, enter 0. If the column direction is defined as the predecessor node, then enter 1 on a_{12} when V_1 points to V_2 similar to Fig. 5a.


V_1 point to V_2
V_1 can reach all of these nodes pass through at most (k-1) items

	1	2	3	4	5	6	7
1	1						
2							
3							
4							
5							
6							
7							

(a)Relation Matrix (A)

	1	2	3	4	5	6	7
1	1						
2	1	1					
3	1		1				
4	1			1			
5	1				1		
6	1					1	
7	1						1

(b) Power of SSIM is k

Fig. 5. Example of A and RX

Step 2: Generate the Reachability Matrix (RX): The Reach-ability Matrix (RX) is used to express whether the relationship between two specific items is direct or indirect. RX can be calculated as:

$$(\mathrm{A}+\mathrm{I}) \neq (A+I)^2 \neq \cdots \neq (A+I)^k = (A+I)^{k+1} \qquad Eq.1$$

$$RX = (A+I)^k \qquad Eq.2$$

"I" is a Unit Matrix. (A+I) to the power of k is RX when (A+I) to the power of k equals (A+I) to the power of (k+1). A power of SSIM = 1 means that a direct relationship exists between discussed items (the connecting route is 1). A power of SSIM = 2 means that the discussed items are connected with a maximum of 2 routes (passes 1 point). In other words, like in Fig. 5b, when the degree of SSIM is k, the items are connected with at most k routes, and pass through at most (k-1) items.

Step 3: Level Partitions: We can find the reachability sets and antecedent sets from every point (element) for the final reachability matrix. A reachability set is defined as a set composed of itself and points that it can reach. An antecedent set is defined as a set composed of itself and points that can reach it. We can obtain the intersection between these two sets from all elements, and get the different elements' levels. If the reachability set and intersection set of the element are the same, then it would be a top level constructed by ISM. Top level elements can be deleted from the set to find the new top level from new set. This tracing step of the top levels is repeated until all of them are found. The level departure can help build the graph of ISM model.

Step 4: Digraph: The final step is to draw a new model from the new obtained elements. First, we draw elements by level. For example, the top-level elements are the root nodes on the attack tree. Thus, they are put on the level first. Then the elements of the second level are located on tree's second level. This is repeated for the third, fourth level, ..., until all elements are added to the attack tree. Next the nodes are connected to the Predecessor node to get the final model while removing the indirect links.

4.2 ISM on Attack Tree

It is necessary to find a relation between elements in the ISM methodology. To reach the goal, the nature of the contextual relationships among the factors must be identified carefully. To analyze the factors, a contextual relationship of "leads to" or "influences" must be chosen. This means that one factor influences another factor. After confirming the relations be-tween parent-child nodes, we can use the ISM method to ex-press an attack tree.

The proposed method includes the following steps (Fig. 6):

1) Indicate the nodes

2) Re-Judge the relationship between nodes and graph the Relation Matrix (A)

3) Generate the Reachability Matrix (RX) through a transformation of the Relation Matrix(A)

4) Level Partitions

5) Digraph

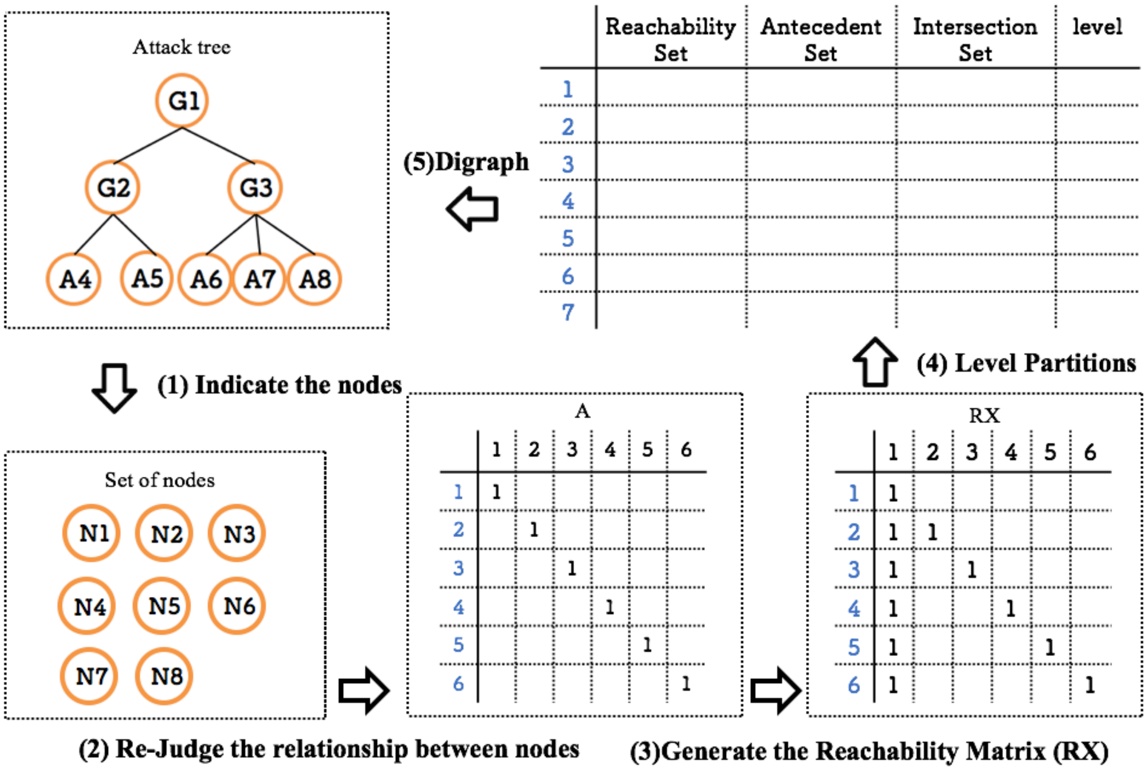


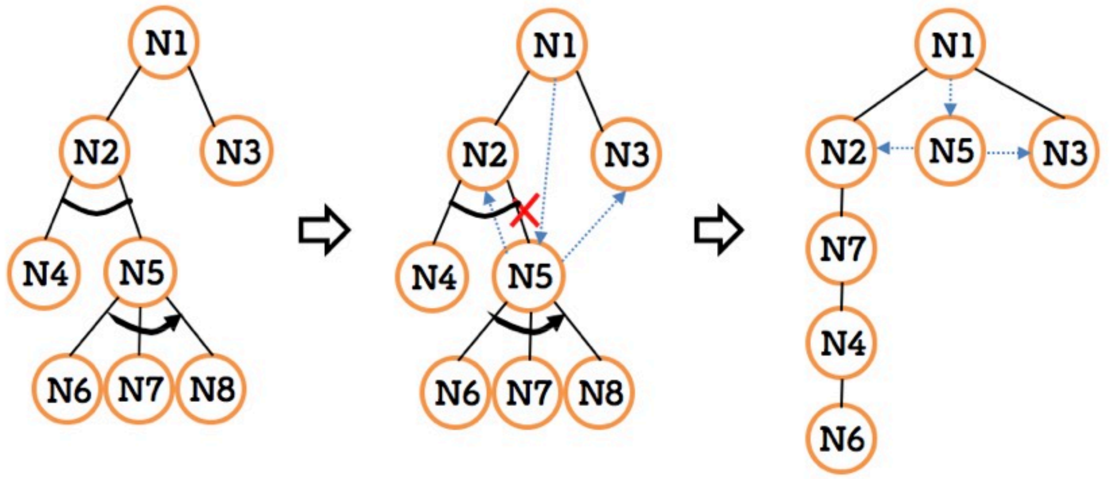
Fig. 6. Overview of our approach

Step 1: Indicate the nodes: In this step, all related nodes are expressed and displayed by symbols. Here Fig. 2 is used as an example. To reduce the computational complexity, the attack tree is separated into two subtrees: *Physical Access* in Fig. 7a and *Invade the Computer* in Fig. 8a.

4.2.1 Subtree Physical Access

Step 2: Re-Judge the relationship between nodes and graph the Relation Matrix (A).

Re-Judge the relationship between nodes: Analyzer should check the relationship between nodes: previously we mentioned that two parent-child relationships, "go through the door" and "reach room2", conflict with the definition *Necessary and Sufficient Relationship between Parent-Child Nodes* in the physical access attack part. However, if these are separated and considered separately, if "reach room2" is true, then "physical access" must be true. Similarly, if "go through the door" or "go through the window" is true, then "reach room2" is true. To let all of the nodes in this subtree satisfy the *Necessary and Sufficient Relationship between Parent-Child Nodes*, we should modify the nodes' relations on an attack tree as follows (see Fig. 7b): Delete the connected line between N2 and N5 and add a line from N1 to N5 because the node "physical access" is true if the node "reach room2" is true. Moreover, add the lines N5 to N3 and N5 to N2 because "go through the window" or "go through the door" is a condition, which must be satisfied to make the node "reach room2" true.



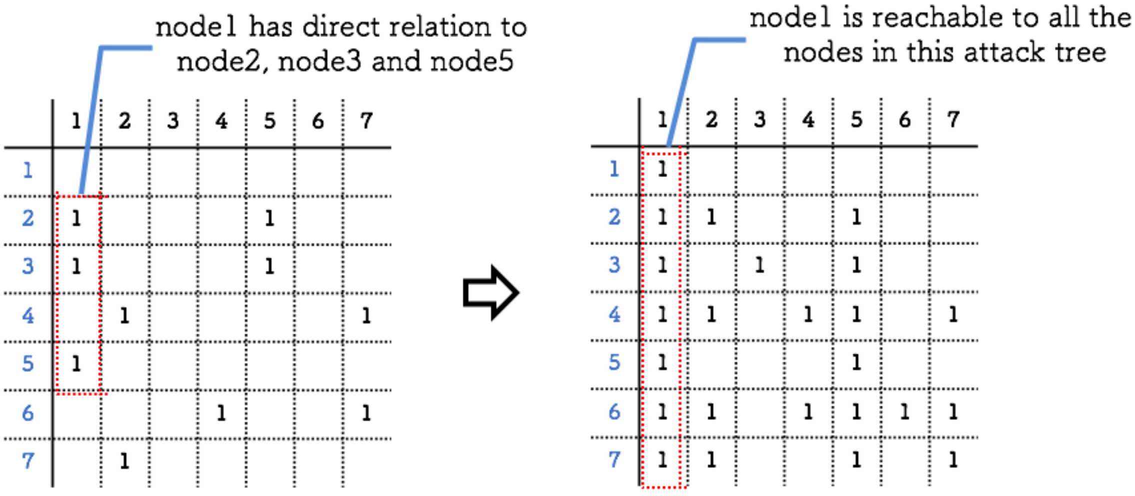
(a)Indicate the nodes (b)Revise the relationship between parent-child nodes (c)Arrange the structure

Fig. 7. Subtree: Physical Access

After revising the relationship between parent-child nodes, arrange the structure of the attack tree to be less confusing. "Reach room2" and "enter room2" are parent-child relations with the same word meaning. Thus, node8 (enter room2) can be removed. We also change the node type AND to a direct vertical structure. For SAND, the later order points are the sufficient conditions for the former order points. Like the at-tack tree in Fig. 7c, node7 (unlock door2) is necessary for node2 (go through the door) to be true. Thereby, node4 (deactivate the camera) must be true to let node7 (unlock door2) be true, and demanding node6 (unlock door1) must be true to let node4 (deactivate the camera) be true.

Graph the Relation Matrix (A): The table in Fig. 8a lists the direct relationships after re-judge the relationship between nodes of sub-tree physical access.

Step 3: Generate the Reachability Matrix (RX): According to Eq. 1 and Eq. 2, the Reachability Matrix can be calculated similar to that in Fig. 8b.



node1 has direct relation to node2, node3 and node5

	1	2	3	4	5	6	7
1							
2	1				1		
3	1				1		
4		1					1
5	1						
6				1			1
7		1					

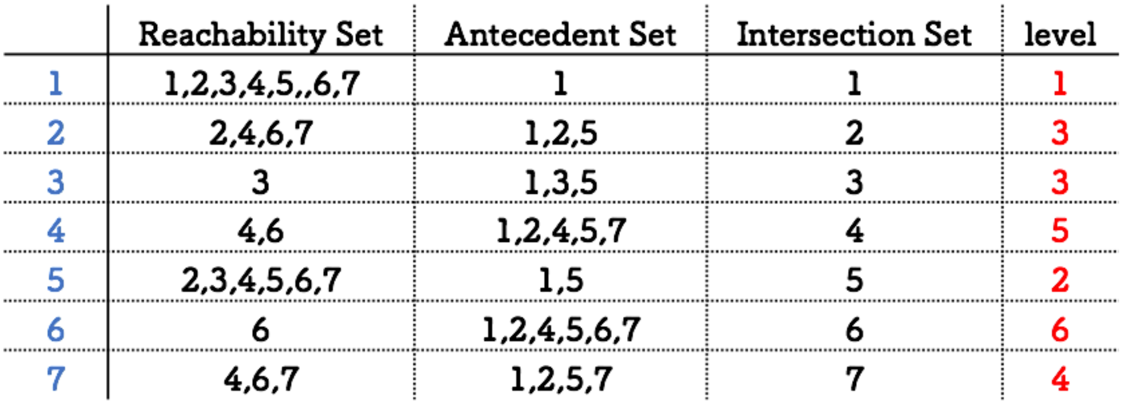
node1 is reachable to all the nodes in this attack tree

	1	2	3	4	5	6	7
1	1						
2	1	1			1		
3	1		1		1		
4	1	1		1	1		1
5	1				1		
6	1	1		1	1	1	1
7	1	1			1		1

(a)Relation Matrix(A)　　(b)Reachability Matrix(RX)

Fig. 8. A and RX of Physical Access

Step 4: Level Partitions: After obtaining the Reachability Matrix for each node, the reachability and antecedent sets are derived. Fig. 9 shows the result using the rule of Level Partitions. The table helps build the ISM model.



	Reachability Set	Antecedent Set	Intersection Set	level
1	1,2,3,4,5,,6,7	1	1	1
2	2,4,6,7	1,2,5	2	3
3	3	1,3,5	3	3
4	4,6	1,2,4,5,7	4	5
5	2,3,4,5,6,7	1,5	5	2
6	6	1,2,4,5,6,7	6	6
7	4,6,7	1,2,5,7	7	4

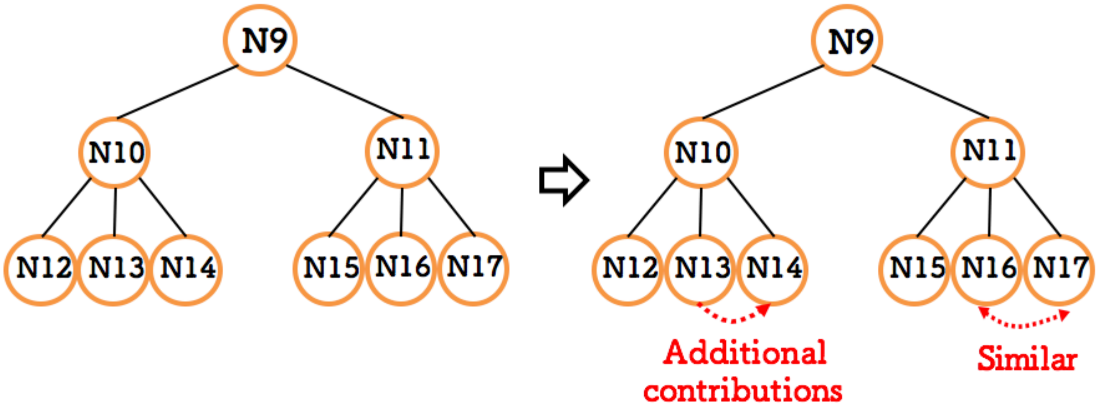
Fig. 9. Level Partitions of Physical Access

Step 5: Digraph: In Fig. 13a, a new model is drawn from the Level Partitions of Physical Access.

4.2.2 Subtree Invade the Computer

Step 2: Re-Judge the relationship between nodes and graph the Relation Matrix (A).

Re-Judge the relationship between nodes: Analyzer should define the relationships between horizontal nodes: In the attack tree, the horizontal relations, which are ignored in the original attack tree, are separated into Additional contribute and Similar (Fig. 10b). As noted in section 2, the attack "WEP crack" can be a condition that makes the node “password crack” at-tack complete in the subtree "invade the computer". It can also be explained that the “WEP crack” can affect the “password crack” attack. Hence, this relationship is defined as additional contributions. Thus, a new path from "password crack" pointing to "WEP crack" can be added between nodes. Besides, the attack of "deceive to download malware" is a similar motion to the attack of "drive by download". This relation is defined as a similar relationship. Because it is unlikely that they would not affect one another, a path is added between them to express the relation.



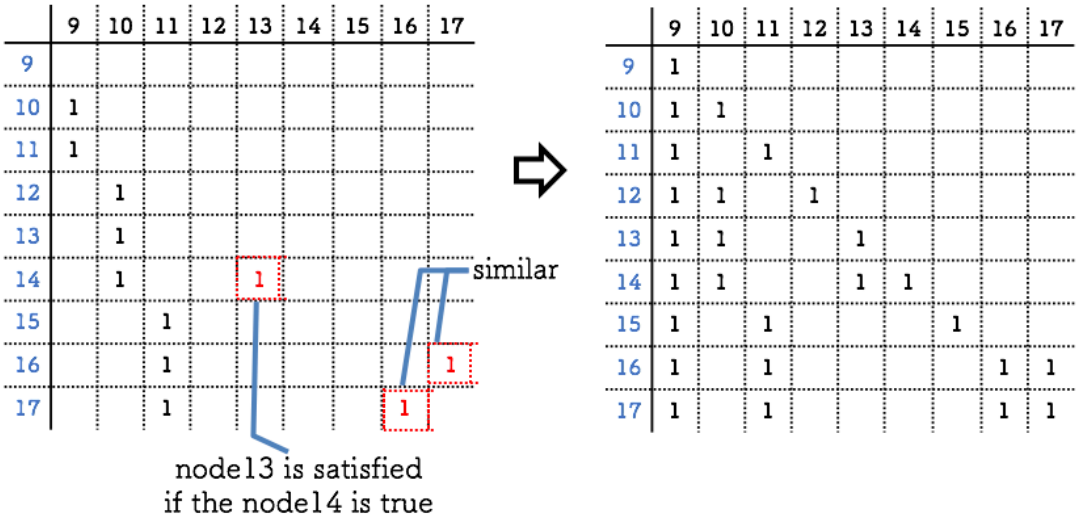
(a) Indicate the nodes

(b)Define the relationship between horizontal nodes

Fig. 10. Subtree: Invade the Computer

Graph the Relation Matrix (A): Similar to the action with subtree Physical Access, Fig. 11a shows the Relation Matrix(A) for subtree *Invade the Computer*.

Step 3: Generate the Reachability Matrix (RX): Fig. 11b shows the calculation of the Reachability Matrix (RX) from the Relation Matrix.



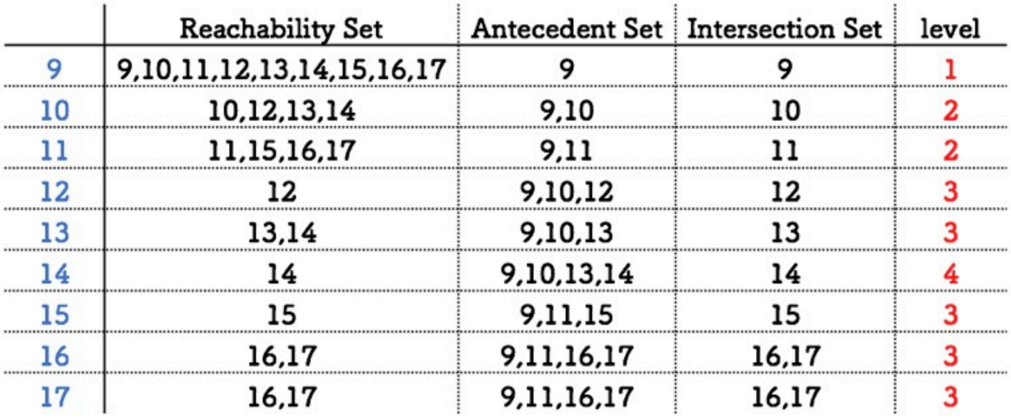
	9	10	11	12	13	14	15	16	17
9									
10	1								
11	1								
12		1							
13		1							
14		1			1				
15			1						
16			1						1
17			1					1	

	9	10	11	12	13	14	15	16	17
9	1								
10	1	1							
11	1		1						
12	1	1		1					
13	1	1			1				
14	1	1			1	1			
15	1		1				1		
16	1		1					1	1
17	1		1					1	1

(a)Relation Matrix(A) (b)Reachability Matrix(RX)

Fig. 11. A and RX of Invade the Computer

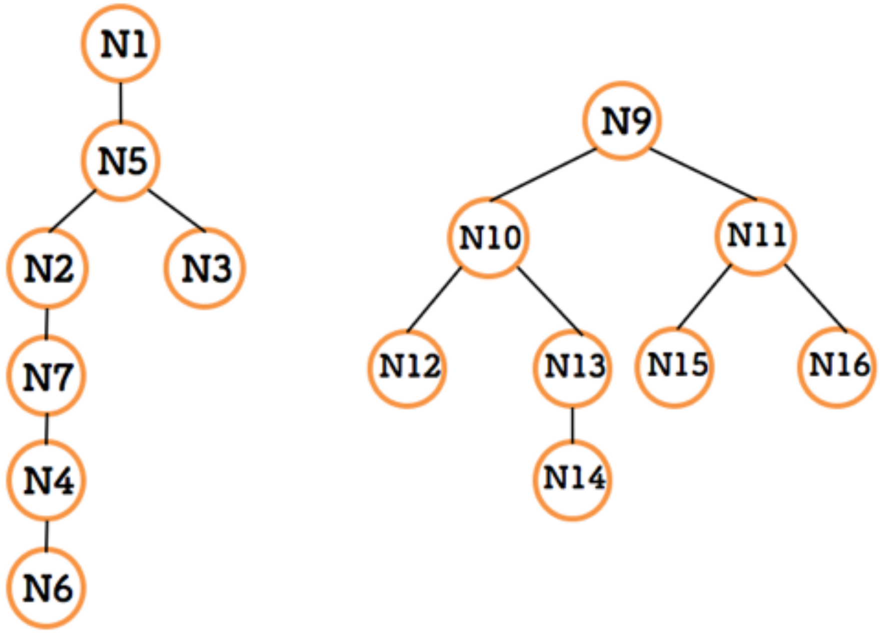
Step 4: Level Partitions: From the Reachability Matrix (RX), the Level Partitions table is drawn. As shown in Fig. 12, two nodes, node16 (disguised as normal software) and node17 (drive by download), are merged into a new node16 and re-named "drive as a normal software" because they have the same result in the *Reachability Set*, *Antecedent Set*, *Intersection Set*, and *Level*.



	Reachability Set	Antecedent Set	Intersection Set	level
9	9,10,11,12,13,14,15,16,17	9	9	1
10	10,12,13,14	9,10	10	2
11	11,15,16,17	9,11	11	2
12	12	9,10,12	12	3
13	13,14	9,10,13	13	3
14	14	9,10,13,14	14	4
15	15	9,11,15	15	3
16	16,17	9,11,16,17	16,17	3
17	16,17	9,11,16,17	16,17	3

Fig. 12. Level Partitions of Invade the Computer

Step 5: Digraph: A new attack tree is drawn from Level Partitions of Invade the Computer (Fig. 13b).



(a) Model of Physical Access (b) Model of Invade the Computer

Fig. 13. Digraph of the two subtrees

5. Discussion

This paper aims to answer three questions (RQs) about our method:

- RQ1: Does revising by ISM ensure that the relations between parent nodes and child nodes satisfy the necessary and sufficient condition?
- RQ2: Does rebuilding by ISM enhance the degree of efficiency of the attack tree?
- RQ3: Is the cost to rebuild a new attack tree by ISM worth the effort?

1) RQ1: We separated this RQ into two aspects. The first is whether the child nodes are sufficient conditions for parent nodes on the attack tree after the ISM revision. When "go through the door" or "go through the window" is true, then the condition "reach room2" must be true. In other words, "go through the door" or "go through the window" refers to "reach room2". It is trivial that the answer for the statement is true. The second is whether the child nodes are necessary conditions for parent nodes in the attack tree after the ISM revision. There are only two ways ("go through the door" or "go through the window") that the statement "reach room2" is true. "Reach room2" must refer to "go through the door" or "go through the window". Combining the two parts provides a conclusion. Based on the complete sets of nodes, the relations between parent nodes and child nodes satisfy necessary and sufficient conditions after revising by ISM method.

However, the complete sets of nodes are affected by other factors such as knowledge limitations of the tree builder or the property of each tree system. Sometimes there is an error due to logical mistakes during the tree rebuilding process. To ensure the relations between parent-child nodes are satisfied with necessary and sufficient conditions on the new attack tree, the relations should be rechecked after revising.

2) RQ2: When we analyze an attack tree, we usually use the probability or cost estimating method to determine the values of possible attacks. For example, Yan Liu [13] mentioned a way to conduct probability analysis on an attack tree. They produced an attack tree whose target is "compromise web plat-form security" with eight child nodes. In this analysis, they referenced the attacked possibility for each path. These values might come from the true data, estimated values over time, cost consuming, feasibility, and the difficulty by specialist. When the number of nodes increases, the work amount to estimate the possibility becomes larger. Take the true data before as an example, the statistics data become increasing large as the nodes increase. On the other hand, when the number of nodes and paths are reduced by analyzing parallel nodes relations, the probability analysis time is minimized. According to the different situations, if an attack tree has large number of nodes, then the relations between nodes are more com-plex. In this content, the enhancing efficiency is much more obvious due to the large amounts of nodes and paths, which can be reduced.

3) RQ3: Due to the range of efficiency caused by the difference in the number of nodes and the difference of analysis methods, quantitative analysis is difficult. The more nodes present, the more that nodes can be reduced. Although the ISM matrix becomes more complex and harder to calculate manually due to the increase in nodes, the matrix can be calculated automatically by programming systems. In other words, it is not an extremely laborious task.

6. Conclusion

Attacks are often used as a model analyzing method in system security analysis due to the complexity of a system. An attack tree may become very large, which leads to two problems: 1) incorrect logic between parent-child nodes and 2) ignored parallel relationships in different branch nodes. The paper presents a definition between parent-child nodes and suggests a method to repair an attack tree with an incorrect relation of parent-child nodes or ignored horizontal relation-ships.

Our method defines the Necessary and Sufficient Relation-ship between Parent-Child Nodes: 1) A child node must be the sufficient condition for parent nodes. 2) A set of child nodes must be the necessary condition for commands of parent nodes. Meanwhile, we define two horizontal relationships: additional and similar. While analyzing an attack tree, we hope that analyzers can start by checking the relations between every child node from the subtree for parent-child logical relations and the parallel nodes from different subtrees. Analyzers should re-judge the relations for nodes and restructure the at-tack tree. Finally, the restructure attack tree should be double checked. Although the revision and verification have some costs, they can verify the accuracy of the attack tree on logical problems, reducing the number of nodes and paths in an attack tree and enhancing the analysis efficiency of the attack tree. Moreover, this method is friendly for beginners as it simplifies the whole system by focusing on the relations between nodes, reducing the burden for users.

Due to the instability in analyzing nodes relations manual-ly, in the future, we would like to find a rapid way to confirm the parent-child nodes' relations. Additionally, we plan to dedicate a program code to the ISM method and make the calculation automatic in order to decrease the rebuilding time and further improve the efficiency of the system.

Acknowledgements

I would like to thank my supervisors, Prof. Hironori Washizaki and Prof. Yoshiaki Fukazawa for their generous support.

I would also like to acknowledge Prof. Takao Okubo of the Institute of Information Security and Prof. Nobukazu Yoshioka of National Institute of Informatics as the advisor of this research

References

[1] Shin-Jer Yang*, "Extended Attack Tree Analysis Method to Assess the Security Risks on the Website," Journal of Information Management, pp.1-38, 2012.

[2] Schneier, "Attack trees, " Dr. Dobb's Journal, 2001.

[3] C. Salter, O. S. Saydjari, B. Schneier, and J. Wallner, "Toward a secure system engineering methodolgy, " in Proceedings of the 1998 workshop on New security paradigms. ACM, 1998, pp. 2–10. B.

[4] A. J¨urgenson and J. Willemson, "Processing multi-parameter attacktrees with estimated parameter values, " in International Workshop on Security. Springer, 2007, pp. 308–319.

[5] A. J¨urgenson and J. Willemson, "Serial model for attack tree computations," in International Conference on Information Security and Cryptology. Springer, 2009, pp. 118–128.

[6] S. Noel, M. Jacobs, P. Kalapa, and S. Jajodia, "Multiple coordinated views for network attack graphs, " in IEEE Workshop on Visualization for Computer Security. IEEE, 2005, pp. 99–106.

[7] S. Mauw and M. Oostdijk, "Foundations of attack trees, " in International Conference on Information Security and Cryptology. Springer, 2005, pp. 186–198.

[8] Higuero M.V., Unzilla J.J., Jacob E., Saiz P., Luengo D, "Application of 'Attack Trees' technique to copyright protection protocols using watermarking and definition of a new transactions protocol SecDP (Secure Distribution Protocol)," Second International Workshop on Multimedia Interactive Protocols and Systems (MIPS)2004], 2004.

[9] Higuero M.V., Unzilla J.J., Jacob E., Saiz P., Luengo D, "Application of 'attack trees' in security analysis of digital contents e-commerce protocols with copyright protection," "Proceedings 39th Annual 2005 International Carnahan Conference on Security Technology, 2005.

[10] Duhoe Kim1 & Dongil Shin1 & Dongkyoo Shin1 & Yong-Hyun Kim2," Attack Detection Application with Attack Tree for Mobile System using Log Analysis," Mobile Networks and Applications, pp. 1-9, 2018.

[11] Hee-Kyung Kong1 · Myoung Ki Hong2 · Tae-Sung Kim3, "Security risk assessment framework for smart car using the attack tree analysis," Journal of Ambient Intelligence and Humanized Computing, Volume 9, Issue 3, pp 531–551, 2017

[12] Jingqi Sun, et al., "Application of Interpretive Structural Modeling in Enterprise Technovation," DBTA, 2010.

[13] Gabriel PETRICA, "Studying Cyber Security Threats to Web Platforms Using Attack Tree Diagrams," ECAI 2017 – International Conference – 9th Edition Electronics, Computers and Artificial Intelligence 29 June -01 July, 2017.

Publications

1. Hua Cai, Hironori Washizaki, Yoshiaki Fukazawa, TaKao Okubo, Haruhiko Kaiya, Nobukazu Yoshioka, “Restructuring Attack Trees to Identify Incorrect or Missing Relationships between Nodes”, Proceedings of The International Workshop on Evidence-based Security and Privacy in the Wild | 2018 (WESPr 2018), 2018.