Integrating cyber attacks within fault trees

Igor Nai Fovino\textsuperscript{a,}\textsuperscript{*}, Marcelo Masera\textsuperscript{a}, Alessio De Cian\textsuperscript{b}

\textsuperscript{a} Joint Research Centre – EC, Institute for the Protection and Security of the Citizen, Ispra, VA, Italy

\textsuperscript{b} Department of Electrical Engineering, University di Genova, Genoa, Italy

\textbf{A R T I C L E  I N F O}

Available online 26 February 2009

Keywords:

Security
Risk assessment
Fault tree
Attack tree

\textbf{A B S T R A C T}

In this paper, a new method for quantitative security risk assessment of complex systems is presented, combining fault-tree analysis, traditionally used in reliability analysis, with the recently introduced attack-tree analysis, proposed for the study of malicious attack patterns. The combined use of fault trees and attack trees helps the analyst to effectively face the security challenges posed by the introduction of modern ICT technologies in the control systems of critical infrastructures. The proposed approach allows considering the interaction of malicious deliberate acts with random failures. Formal definitions of fault tree and attack tree are provided and a mathematical model for the calculation of system fault probabilities is presented.

\@2009 Elsevier Ltd. All rights reserved.

1. Introduction

Modern industrial systems aggregate typical ICT systems with real time control systems. The interconnection between these different networks makes the cyber security of industrial computer networks extremely important for the protection of critical infrastructures such as electric power plants, aqueducts or transport networks. In fact, a security fault in any part of such systems, including office computers, may allow an attacker to gain access to the core of their control sub-systems, allowing him to interfere with their correct behavior, potentially causing critical damages at different levels. Several studies have been conducted, showing how it is possible, by taking advantage of ICT security lacks, to seriously damage critical infrastructures like power plants, gas pipelines, etc. See for example\cite{19–22}.

In the scientific literature, several methods have been proposed for the assessment of the safety/security exposure of complex systems. Historically, one of the well known analysis techniques used to assess safety/security breaches is the fault-tree analysis. Such techniques are strongly oriented to the concept of system/component reliability.

The introduction of ICT technologies in Critical Infrastructures originated a new layer of complexity in the analysis of the security/safety of such systems, influencing the type of security faults that might happen, the granularity of interdependency and complexity of description. The safety and security issues of traditional infrastructures did not change much during the lifecycle of the system. Therefore fault trees built during the design of the system were valid, with little adjustments, for its whole life cycle. The same cannot be said about today’s infrastructures. The physical configuration of a modern system may remain the same for all the lifecycle, but the cybersecurity issues evolve continuously, as new vulnerabilities are discovered almost every day. Therefore, a method which takes into account the new problems is needed to analyze the security of today’s systems.

In 1998 Schneier\cite{11} introduced the attack-tree paradigm for the description of the process by which an attacker can exploit successfully a target system. An extended version of such paradigm has been presented by Masera and Nai\cite{9} capturing both the probabilistic and semantic meaning of an attack process associated to a target system. In this paper we claim that, in order to assess the safety/security properties of a target critical infrastructure, it is necessary to assess both the critical event chains caused by random events and by malicious cyber events.

A possible solution to such challenge is the integration of fault-tree analysis and attack-tree analysis.

This paper is organized as follows: after an overview of the State of the Art in these fields, we propose a new formalization of attack tree (sustained by our previous results) focused on the concept of malicious event and attack goal. Moreover we introduce the new concept of macro-attack tree allowing the creation of a multi-layered view of the attack process.

In light of such definitions, we show then how the concept of “attack goal’ could be formally treated for being integrated into fault trees. Finally, both in formal and practical ways, we present the technique for merging a set of attack trees with a set of pre-existing fault trees. It is shown how it is possible to use such new integrated information (expressed in term of probabilistic events) in the fault-tree analysis.
2. State of the Art

Fault trees, first presented by H.R. Watson of Bell Laboratories in 1961 [13] for reliability and safety analysis of the Minuteman intercontinental missile and later developed by Boeing, have been widely used in the last decades in the field of complex system safety and are now the most well developed and applied technique for the analysis of faults. The instrument of fault trees has improved introducing methods for their fast building and calculation. A summary of the most important papers on the subject can be found in [5]. A generalization of the concept of fault tree is presented in [7]. Recently, fault-tree analysis has been applied to study cybersecurity problems: in [6] the authors apply fault trees to analyze the security requirements of a software intrusion detection system, underlining the usefulness of fault trees in software design; Brooke and Paige [2] show how fault trees can be used to design not only safety-critical systems but also security-critical systems.

Attack trees are typically used in order to describe attack processes. The most promising approach for describing in a complete manner the attacks pattern is known as Graph Based Attack Models [15]. It is possible to categorize such models into two big classes:

- The petri net-based models
- The attack trees models.

The Attack Net Model introduced by McDermott [16] in which the places of a Petri Net represent the steps of an attack and the transitions are used to capture precise actions performed by the attackers falls in the first class of methods.

The second approach (attack trees), originated from the world of fault analysis, adopts a tree representation of the dependencies among component of a system, vulnerabilities and actions performed by attackers. In this context, Bruce Schneier [11] proposed to use a technique based on the use of expansion trees to show the different attacks that could affect a system. Attack trees can be used to capture the steps of an attack and their interdependencies. Such an approach has been largely used and improved. For example Daley et al. [17] have proposed to introduce a layering approach (stratified node topology) in the attack tree design, in order to separate the attack tree nodes based on functionality (event level, state level, etc.). Moreover, in such a context, recently, Jain et al. [18] have introduced an approach based on the concept of vulnerability topological analysis, allowing, starting from the combination of modeled attacker exploits, to discover attack paths. Moore et al. [8] introduced the concept of reusability of trees through the organization of patterns of commonly performed attacks. Finally, Masera and Nai [9] have proposed a multi-dimensional attack tree representation for matching abstract attack trees directly onto target systems.

Papers dealing with the modeling of system reliability by means of fault trees do not normally take into account malicious attacks, but only random faults; conversely, most papers describing attack modeling techniques do not consider the effects that the achievement of the attack goal can cause to high level services of the system.

Some recent works underline the necessity to integrate the information about attack patterns provided by attack trees and the modeling of faults given by fault trees. In [12] the authors introduce the concept of “aggregated security” of a system, which takes into account all the possible failures of components and services, including the ones caused by threat agents, in order to evaluate the overall security of the system but without giving details on how quantitative analysis of attacks can be used in the fault-tree analysis.

In [1] the authors introduce a modified attack tree with leaf nodes which can take into account environmental causes that can influence an attack pattern, but this approach cannot be directly integrated into fault-tree analysis processes.

3. Preliminary definitions

Fault trees and attack trees were developed in different fields (reliability analysis and computer security respectively), with different formalisms; a unified presentation of their definitions and properties, is then needed in order to achieve the goals of the research presented in this paper.

In this paragraph, formal definitions of the main concepts involved in the paper are presented.

3.1. Events

For linking security incidents with random failures the approach proposed in this paper makes use of the concept of Event [14]. An event \( E \) in the integrated view proposed herein refers both to accidental events (e.g. the failure of a component) and to actions performed by malicious actors against the system components (e.g. an action meant to cause damage). For the quantitative analysis, each event should be characterized by a probability value \( (PE) \) measuring the likelihood of its occurrence.

3.2. Fault tree definition

In the following it is given a formal definition of fault trees \( FT_k \) as logical structures which allow expressing the relationships and dependencies between a high level \( top \ event \), and lower level events: each of these lower level events being either a basic event, not dependent on others, or an intermediate event, dependent on other events. These definitions are then used for the integration with attack trees.

In this paper, \( e_i \) represents a single event, \( EU = \{ e_i \} \) is the universe of events and \( ES = \{ e_i \in EU, e_i \in FT_k \} \) is the set of events that belong to the fault tree \( FT_k \).

The relationships between events are modeled using standard logic gates; from now on, \( p_k \) will denote a logic gate, \( PU = \{ p_k \} \) will be the universe of logic gates, and \( P_k = \{ p_k \in PU, p_k \in FT_k \} \) is the set of logic gates that belong to the fault tree \( FT_k \).

A logical gate \( p_k \) is characterized by \( n_k \) inputs \( IN_k (p_k, j = 1, ..., n_k) \) one output \( OUT(p_k) \) and a logical function \( f(p_k) \) such that \( OUT(p_k) = f(p_k(IN_1, p_k, IN_2, p_k, ..., IN_{n_k}, p_k)) \).

Events and gates are connected to each other through directed edges; an edge \( d_{a,b} \) connects two nodes \( n_a \) and \( n_b \).

The set \( DU = \{ d_{a,b} \} \) is the universe of directed edges, and \( D_k = \{ d_{a,b} \in DU, d_{a,b} \in FT_k \} \) is the set of directed edges that belong to the fault tree \( FT_k \).

We can define a fault tree as follows:

**Definition.** A fault tree \( FT \) is a directed acyclic graph defined by the tuple \( \langle E_S, P, D_k, TOP \rangle \).

The union of the sets \( P \) (logical gates) and \( E \) (events) represents the nodes of the graph; \( D \) is a set of directed edges, each of which can only connect an event to the input of a logical gate or the output of a logical gate to an event, while direct connections between logical gates or between events are forbidden:

\[
D_k \subset \{ (n_h, n_i) \in \{ (E_i \times P) \cup (P \times E_i) \} \}
\]

(1)

where \( (n_h, n_i) \) represents the edge between elements \( n_h \) and \( n_i \).
A top event $TOP_i$ is an event of the fault tree $FT_i$ that is not the input of any logic gate, i.e. there are no edges that come out of the top event:

$$\exists d_{ij} \in D_i : d_{ij} = (TOP_i, p_i) \quad \forall p_i \in P_i$$

(2)

A basic event $BE_i$ is an event of the fault tree $FT_i$ that is not the output of any logic gate, i.e. there are no edges that enter the basic event:

$$\exists d_{ij} \in D_i : d_{ij} = (BE_i, p_i) \quad \forall p_i \in P_i$$

(3)

Moreover, let $D_i : d_{ij} = (BE_i, p_i) \quad \forall p_i \in P$ be the set of the directed edges from a basic event $BE_i$ to the set of logic port $P$. The cardinality of such set must be equal to 1. In other words, only one edge comes out of a basic event, i.e. if different events depend on a basic event $BE_i$, then there are multiple instances of $BE_i$ in the tree.

In the scientific literature [7], generalized formulations of the fault tree concept have been proposed. In these new formulations, there can be several top events $TOP_i$, $j = 1, \ldots, n_{top}$, and there can be more than one edge going out of a basic event. The definition of fault tree presented above takes into account also the generalized formulations, since also in that case the structure is still a directed acyclic graph. Therefore, in the following paragraphs, even if a traditional fault tree structure will be used for the integration with attack trees, everything could be reformulated using generalized fault trees.

3.3. Attack tree definition

An attack tree is a particular graph that describes the steps of an attack process. As explained before, the concept of attack tree was introduced to describe and share information about attack patterns. Hence typically they are not usually focused on a target scenario (as it is typically the case for fault trees). On the other hand, from the risk assessment perspective, it is useful to have attack trees focused on target scenarios in order to produce more precise and detailed analyses. For this reason, we have chosen to adopt an extended version of the attack-tree paradigm, presented by Masera and Nai in [9], which allow to map abstract attack trees over target systems, taking into consideration also information like “attacker resources”, “attacker motivations”, etc.

Every attack has a final scope or a final motivation. For example, the final scope of a Denial of Service attack against a Web-Server could be to avoid to a set of users the access to the data. Usually in every attack process the attacker, in order to reach his malicious goal, take advantage of some system vulnerability. For this reason an attack tree must also contain the description of the vulnerabilities used. As in [9], we define vulnerability as follows:

**Definition.** A vulnerability can be described by a tuple $\langle Name; Type; Description; Vulnerability reference; list of affected components; Countermeasures list; Sev; Likelihood; resources required \rangle$ where, see [10], Sev is an index representing the severity of the vulnerability and Likelihood represents the subjective probability of the occurrence of the vulnerability. The use of subjective probabilities in the IT security and IT risk assessment fields is quite common [24]; according to De Finetti [23], the state of knowledge corresponds to a “personal belief”. In the case of the vulnerabilities, the “a priori” likelihood of their presence in a system is based on a subjective evaluation of the analyst which carries out the security assessment, or, in other words, it is based on the knowledge, experiences and security perception of the analyst. Every time such knowledge changes, the estimation of the subjective probability associated to vulnerabilities changes.

Given the definition of vulnerability, we can then define $V = \{v_j\}$ as the Universe of the known vulnerabilities.

We have now the bricks for composing an attack tree. Formally, we define an attack tree $AT_k$ as a tuple $AT_k = \{g_{ik}, O_{ik}, A_{Si}, V_i, R_i\}$ where $g_{ik} \in G$, $O_{ik} \subseteq O$, $A_{Si} \subseteq AS$, $V_i \subseteq V$ and $R_i$ is the set of relationships among the elements of the tree.

In this way, by construction, every attack tree has only one main (or principal) goal. The relationships between assertions, operations and vulnerabilities are established by logical gates (see the previous section related for the definition). The only compositional constraints for the combined use of gates, assertions, operations and vulnerabilities are:

1. The output of a logical gate has to be an assertion (except for the case expressed in the following).
2. The output of the top logic gate has to be the goal of the attack tree.

The structure of an attack tree can also be seen as the composition of a set of low level, attack trees. As defined in [9] we can call them Micro-Attacks, or, if composed just of one attack layer (assertions-components-vulnerabilities, logic gate and goal), Atomic Attack trees. When an attack tree results from the composition of several Micro/Atomic attack trees, it is called macro-attack tree (we give the definition of macro-attack tree in the following section).

On this basis, we are now able to introduce the concepts needed for the integration of fault trees and attack trees.

4. Trees integration

A fault tree describes how a set of events can concur in order to cause a certain Top Event. In the field of System Security fault trees are usually used to describe the “race conditions” of a system, i.e. the combinations of events which can, in some way, damage its functionalities. In the same way, attack trees are used to describe the steps an attacker must follow in order to successfully exploit some functionalities of a system. It is then evident how the two paradigms (attack trees and fault trees), originated in two different worlds (ICT Security and System Reliability), can be integrated in order to obtain a more powerful and rich picture of the exposition of a target system to failures and damages. Most importantly, this integration can help in understanding how malicious attackers can take advantage of the failures of components for deploying their hostile actions.

The scientific literature related to Fault Trees Analysis is, nowadays, mature. In the reliability and safety fields there exist a large number of examples and case studies. We show here how it is possible to take advantage of a set of pre-existent fault trees related to a target system, from completing the security analysis derived from attack trees.
Remembering the definition given in the previous section, a set $EU_k$ represents the set of the events captured by a target pre-existing fault tree $FT_k$.

Let us consider now the attack trees; we can define the set $A = \{at\}$ as the universe of the known Micro and Atomic attack trees. Moreover, with respect to the definitions given in the previous section, we define the set $G$:

$$G = \{goal \ at, \ \forall at \in A\}$$

as the set of the attack goals (the notation “$a$B” is common in computer science to denote a property $a$ associated to an object $B$). In that case, $goal \ at$, means “the goal of an attack tree at”.

A goal represents the final result of a successfully exploited attack. However, this final result can be interpreted as an event which would have happened the attack success.

The ultimate scope of an attack is usually to perform an “unauthorized operation” on a system, i.e. to produce an unwanted event. We note that the events described in fault trees are events which, when occurring, could cause some damage. We propose to enrich fault trees by considering malicious actions (modeled by attack trees) that can cause some of its basic or intermediate events. Conceptually, it would mean that an attacker can take advantage of some failures in the system for ultimately causing the Top Event.

Then we propose the equation $\text{Attack tree goal} = \text{Fault tree event}$ showing the fact that a top goal of an attack tree could coincide with an event contained in a fault tree.

The only difference between the event contained in the fault tree and the top goal of the attack tree is its origin. The attack process is always originated by a malicious agent and regards ICT infrastructures (although a similar process could possibly be used for physical attacks, but this lies beyond the scope of this paper), for physical attacks, but this lies beyond the scope of this paper).

Starting from these arguments it is possible to easily integrate the information contained in the attack trees with the information contained in the fault trees. In order to do this, however, we need to specify a new structure allowing the grouping of all attacks that share the same goal.

Let $A_k$ be the set defined as

$$A_k = \{at : \text{goal} \ at = g_k, g_k \in G, \ \forall at \in A\}$$

$A_k$ represents the collection of all the known attack trees having the same final goal in common. In order to have all the bricks for building the new structure, we define a special logic gate named $\text{TOP}_{\ OR_k}$. Such a gate is an OR gate with $n_k$ inputs and 1 output where:

$$n_k = |A_k| : \text{input, TOP}_{\ OR_k} = at_i, \quad i = 1 \ldots n_k, \text{output} \ \text{TOP}_{\ OR_k} = g_k, g_k \in G$$

**Definition.** The structure composed by $g$, $\text{TOP}_{\ OR_k}$ and its related inputs is a macro-attack tree $MA_k$.

A macro-attack tree summarizes all the known ways by which it is possible to reach a certain malicious goal.

**Definition.** Let $MA = \{MA_k\}$ be the universe of the known macro-attacks, a fault tree $FT_k$ can be integrated with information derived from the attack tree universe if and only if

$$\exists e_i \in EU_k : e_i = \text{goal} \ MA_j, \quad MA_j \in MA$$

In other words, a fault tree can be integrated with a set of attack trees only if there exist goals of attack trees which are events of the target fault tree.

If this is the case, it is possible to merge the target fault tree with the macro-attack, as follows:

1. The sub-tree ($e_{i\ subtree}$) which, in a bottom up view, originates the $e_i$ event, is detached from the fault tree.
2. An OR logical gate (merge gate) with two inputs ($A$ and $B$), is connected to the $e_i$ event. From a logical point of view, we say that the event $e_i$ is the output of the merge gate.
3. The macro-attack $MA_j$ is connected to the input $A$ of the merge gate. Remembering that in our approach we consider the goal of an attack as an event which happens if the attack is successfully exploited, in this way we have preserved the composition rule of a well formed fault tree (the input and the output of every logic gate must be an event.
4. The goal of $MA_j$ is modified into “successful attack causing $e_i$.” In such a way we avoid to have as input and output of the $\text{merge} \ gate$ the same event.
5. The virtual event “Original $e_i$” is introduced and connected to the input $B$ of the merge gate.
6. The sub-tree $e_{i\ subtree}$ is then connected to the virtual event.

Adopting this procedure we have formally obtained a new entity which is a fault tree enriched with information derived from the universe of the attack trees. We call such entity an extended fault tree (EFT).

5. Trees integration: quantitative analysis

Each event of a fault tree is characterized by a probability value: once the probabilities associated with leaf nodes are known, it is possible to calculate the probability of higher level events. Since, in our case, some of the leaf nodes of the fault tree coincide with the goals of one or more attack trees, it is necessary to develop a mathematical formulation of attack trees that enables us to calculate a probability value associated with the goals.

Attack trees will be represented using the notation introduced in [9], where there are three different kinds of nodes: vulnerability nodes, denoted in the following figures by ellipses, which represent vulnerabilities that an attacker may exploit; operation nodes, denoted by hexagons, which represent actions performed by the attacker and/or by an authorized operator, and assertion nodes, denoted by rectangles.

5.1. Some considerations on the treatment of probabilities

In the model we have adopted [9], the uncertainty about the occurrence of the top event of attack trees is associated with a subjective probability. This follows modern ICT risk assessment models [24–26], in which, considering the lack of objective and significant amount of data about attack statistics, the uncertainty is expressed on the basis of expert opinion [3,4]. The probability of an attack goal can be seen in the context of a fault tree as the probability of a potential single event conditional to some assumption: in our case that assumptions are related to considerations like [27]:

- Motivation of Threat Agents.
- Resources of the Threat Agents.
- Environmental conditions.
- Subjective probabilities associated to the individual elements constituting the attack tree (assertions, operations, vulnerabilities).
In [9,10], foundations are provided in order to map abstract attack trees over a given system taking into account information like “attacker resources”, “attacker motivations”, etc.

5.2. Node probabilities

In order to calculate the probability associated with the goal of the attack tree, we have to assign a probability value to all nodes in the tree. Each node can belong to one of the three classes defined above: operations, vulnerabilities and assertions: a definition of the probability value is needed for each class.

Let \( O_i \) be an operation. The probability \( P_{O_i} \) associated with \( O_i \) is equal to the likelihood, assigned by the analyst on the basis of his experiences and knowledge, that such operation is executed with equal to the likelihood, assigned by the analyst on the basis of his experiences and knowledge, that such operation is executed with success by an attacker of an authorized operator. It will be indicated as follows:

\[
P_{O_i} = Pl O_i
\]

Let \( V_i \) be a vulnerability. The probability \( Pl V_i \) is defined as the likelihood of the existence of the vulnerability in a specific component/subsystem; thus, we can define the probability \( P_{V_i} \) associated with \( V_i \) as

\[
P_{V_i} = Pl V_i
\]

Let \( A_i \) be an assertion. In order to consider \( A_i \) true, the logical expression \( LogicExp A_i \) must be validated (for a formal definition of attack tree assertions, please see [10]). Therefore, the probability associated with \( A_i \) can be defined as

\[
P_{A_i} = p(LogicExp A_i)
\]

5.3. Logical operators

The nodes of an attack tree are connected through logical gates, which have an assertion as output. In this paper, we consider only static attack trees, i.e. attack trees in which the time variable does not appear. Therefore, gates such as NAND (priority AND) will not be used even if, in principle, the approach used here can be adapted for use with dynamic attack trees; only the AND and the OR gate will be treated in this paper. As known, the output of an AND gate is equal to 1 only if all its inputs are equal to 1. Let us suppose \( AND \) is an AND gate with \( n \) inputs \( IN_1, AND_n, 1 < k \leq n \) and output \( OUT_1 \). For simplicity sake, in the following formulas, the inputs will be denoted by \( IN_k \). Let \( Pin(k,i) \) be the probability associated with the input \( IN_k AND \), and \( Pout AND \) be the probability associated with the output of \( AND \).

If the inputs to the AND gate are mutually independent, the probability associated with the output can be calculated as follows:

\[
P_{out \text{ AND}_i} = \prod_{k=1}^{n} P_{in}(k, i)
\]

5.4. Top event probability calculation

In this paragraph it is shown how, using the integrated view of fault trees and attack trees, it is possible to calculate the probability of occurrence of the top event of a fault tree in which one or more events are the outcome of a malicious attack.

Let us consider the fault tree in Fig. 1, which illustrates how an undesired high level event, labelled “Top Event” is logically related with lower level events (E1–E7). The top event considered in the example is the release of a toxic substance into the environment by a chemical plant; this event can happen if a pipe breaks because of an overpressure and the failure of both automatic protection system and remote shutdown commands, in conjunction with the failure of the containment system. In the figures of this paragraph, underlined labels will denote probabilities which are known at the beginning of the procedure, while
The probabilities of events E4, E5, E6 and E7 are initially known; the event E3 is the possible outcome of an attack A. In order to perform the calculations of the fault-tree analysis, it is required to calculate (or at least estimate) the probability of the occurrence of event E3. This goal can be achieved by means of a probabilistic Attack-tree analysis which makes use of the concepts and formulas presented in the previous paragraphs. Let us suppose that the attack tree describing the possible attack pattern that causes the event E3 is the one in Fig. 2. The event marked as E3 is related to the failure in the remote shutdown process. This event can be caused by an attacker in several ways. The most evident is an attacker trying to cut off the network connection of the remote server, which is then unable to deliver the proper shutdown command. For example, if we suppose that the server is based on Kernel 2.6 without patches and with the IPtable service active, an attacker –by simply sending some ad hoc malformed packets – would be able to inhibit any network connection (by exploiting a well known bug contained in IPtable on Kernel 2.6).
systems). On the other hand if we assume that the server is accessible from networks with “zombies sub-network”, an attacker controlling the zombies can shut down the server network connection by simply performing a DDoS consuming the bandwidth.

It is possible to calculate the probability associated with the attack tree goal exactly in the same way as it is done for the fault trees.

The only difference between the attack tree shown in Fig. 2 and a standard fault tree resides in the “Operation” node: while in a fault tree initially we only know the probabilities of occurrence of the leaf (lowest level) nodes, in an attack tree we also take into account information about the probability that an operation can be really performed by the attacker. Since in fault-tree analysis initially the known probabilities do not include the intermediate nodes, it is necessary to transform the logical structure of Fig. 1 into an equivalent structure where only leaf node probabilities are known (Figs. 3 and 4).

Let us suppose we have calculated the probability $P_{O1}$ (i.e. the probability of the pre-conditions of operation 3). In order to make the structure of the attack tree identical to that of a fault tree we can transform the node “Operation” in the way shown in Fig. 3.

The Operation node (with probability $P_{O1}$) is substituted by an AND gate $AND_{O1}$ with 2 inputs: the first input consists in the original input of the Operation node; the second input is an assertion with a probability equal to $P_{O1}$ and represents the probability that the attacker decides to perform the operation.

Once all the Operation nodes with an input have been transformed, the attack tree will become as shown in Fig. 4.

Once the Operation nodes have been transformed, it is possible to perform the integration of fault tree and attack tree as it was described above; the resulting logical structure for the example is illustrated in Fig. 5.

We have then integrated the attack tree which potentially could cause the event $E3$ with the pre-existent fault tree, considering then in the final evaluation of the TOP_Event probability also the contribution given by cyber malicious actions.

5.5. Modeling considerations

In the previous section we have showed how to practically obtain an EFT starting from a fault tree and a set of attack trees. If,
from a side, such a process is clearly defined in term of “what to do, how to join two trees etc.”, a not negligible role, in order to obtain a good EFT, is played by the modeling process. Let us consider the following example: we have a fault tree \( F \), one of its event, the event \( E_j \), fit with the attack goal of a macro-attack tree \( AT_1 \). According to what we have described, we proceed in the integration of the \( AT_1 \) into the fault tree \( F \). A superficial analyst could think that the obtained EFT is now complete. By construction, the macro-attack tree \( AT_1 \) is composed by several mini-attack trees. Each one of these mini-attack trees could provoke some “side effects” which can have basically an impact on the system under analysis. More specifically some of these side effects could be identical to some other event \( E_n \) of the considered fault tree \( F \). In such a case, the EFT obtained just by mixing a fault tree \( F \) and a macro-attack \( AT_1 \), has to be considered badly modeled, since side effects have not been taken into consideration. The proper modeling approach should be then the one in which also the sub-goals of the involved micro-attack trees are taken into account. In other words, if a sub-goal \( G_k \) is equal to some other event \( E_k \) of the fault tree, then the integration procedure should be repeated taking into consideration the subtree having as “head” \( G_k \) and the FT event \( E_k \). In this way, also all the side effects caused by the exploitation of a macro-attack tree against a target system can be taken into account in the creation of a new EFT.

6. Conclusions

The protection of Critical Infrastructures (CIs) against security threats and vulnerabilities is one of the big challenges of the modern era. Recently, the exhaustive use of ICT technologies in these infrastructures has augmented the exposure of CIs to new malicious threats. These threats thrive on the easy access to communication networks and the use of standards technologies. In this way, systems that used to be rather unknown from the point of view of their structure and operations, and almost impossible to reach to external actors, are now at risk of security threats. The traditional approach, successful in the treatment of safety and reliability issues, ignore cybersecurity. Now there is an urgent need to complement previous analysis methodologies with the consideration of security factors. In this paper, after a presentation of the main issues related to fault-tree and attack-tree analyses, we proposed a technique for the integration of their analysis structures. It is shown how it is possible to integrate attack trees into a pre-existent fault-tree in order to extend the usability of the results of a traditional risk analysis with the consideration of potential malicious attacks. The method presented here is supported by proper models for integrating fault trees and attack trees. Such approach promises to be helpful not only in the analysis of the risk exposure of ICT systems, but even in
the discovery of new complex attack pattern profiles. For sake of completeness, we have to say that the presented approach considers only “static attack trees”, i.e. attack trees in which the time dynamics are not taken into account. This is mainly due to the lack of a large knowledge base in the ICT security community about complex attack temporal conditions parameters and values. However, at the moment we have started to integrate the model presented in this paper with Binary Decision Diagram (BDD) techniques and Monte Carlo simulations, in order to capture also such kind of aspects. Moreover we plan to integrate the proposed process in a comprehensive risk assessment methodology. In particular, we will make reference to the process InSAW (Industrial Security Assessment Workbench), partially described in [10].

References


