17

Analysis of Timing Requirements for Intrusion Detection and Prevention using Fault Tree with Time Dependencies

Pawel Skrobanek and Marek Woda
*Institute of Computer Science, Automatic Control, and Robotics
Wroclaw University of Technology, Wroclaw,
Poland*

1. Introduction

In-depth analysis of an attack strategy enables possibility to prevent it or when it is inevitable to minimize its adverse effects. Intrusion detection systems (IDS) base their operation on the built-in patterns of various attack strategies. Aforementioned strategies can be represented by different means like: augmented goal-tree [6], attack trees [17] (originated from on fault trees), attack graphs [14], or augmented software fault trees [4].

In augmented goal tree representation [6], the attack is expressed by sequences of logically related steps. The root of this tree is the goal of the attack, e.g., “Modification of a file”. The sub-goals are associated with the roots of the sub-trees. The basic constructs are the OR, AND, Ordered-AND constructs. For example, in order to achieve the sub-goal represented as the root of the Ordered-AND construct, all sub-sub-goals have to be reached in required order.

Fault tree analysis (FTA) [3], which is a base of the attack tree [17] analysis, is a deductive probabilistic assessment technique. The FTA is the backward approach. In the fault tree, the root is associated with the top event being the hazard, e.g. “The Intrusion takes control over the Victim”. Then direct causes of the hazard are considered. Next, the causes of the above causes are analyzed. Hence, different ways in which the hazard can occur are investigated. The FTA can be used to determine the following: minimal cut sets of faults that cause a hazard, probabilities of the hazard and faults. Therefore, the FTA can be used in identification which events are critical and should therefore be subjected to monitoring.

Traditional Fault Trees (FT) are widely criticized [14] due to many, widely known drawbacks, like inability to model multiple attack attempts, time dependencies, or access controls as well as for luck of modeling cycles.

In the attacks graphs [14], nodes symbolize the class of machines the attacker accessed and as well user level of privileges. The arcs are labeled by attackers’ activities. By assigning probabilities of success on the arcs, one can identify the attack paths with the highest probability of success.

Augmented software fault trees [4] were defined in order to overcome disadvantages of the classical fault trees. In this approach, a trust, a context, and temporal orderings can be defined. The trust relationship expresses that some members of a distributed system trust other members of the system. Context describes which subsets of intrusive events occur in some context. The activities of attackers and network are time dependent. Event and conditions involved in an intrusion often must occur in a particular order. In order to express the temporal orderings, the interval temporal logic [1] is applied. In this logic, the structure of time is a simple linear model of time, i.e., there is one past and one future only. Therefore, the attack scenario is a deterministic one.

The fault tree with time dependencies (FTTD) [9], [10] can express non-deterministic attacks too. In the chapter, FTTD are used to describe attacks with emphasis put on timing properties. In the FTTD, event duration times, delay times between the cause events and the result event can be described by the time intervals given by its minimal and the maximal lengths. FTTD are used in verification whether the IDS reacts sufficiently quick on the attack in order to avoid the results of the attack.

The fault trees could constitute a basis for increased security awareness in any organization by supporting IT infrastructure audit preparation by drawing conclusion out of FT analysis and based on that security survey creation [20].

The structure of the chapter is the following. In, Section 2, models of two attacks (SYN-Flood called "The Victim trusts the Intruder" and generic "The Intruder has access to data in server") are described. In Section 3, FTTD in general, and FTTD for this attack are presented. Then the analysis of timing properties of this attack using the FTTD is given. In Section 4, authors exemplify protections against the attacks and present methods that increase security awareness and facilitate enforcing the desired security level. In the last section, conclusions are being stated.

The details of the TCP/IP protocol are given in [7], [8], [12], [15] and [16].

2. System description

We base our analysis on two cases, exemplified by following attack examples:

1. "The Intruder has access to data in server",
2. "The Victim trusts the Intruder".

2.1 Example #1: "The Intruder has access to the data stored on server"

Analyzed system is illustrated by Figure 1. There are two possible situations shown:

a. The intruder have access to the terminal T, onsite (within company's network) or uses trusted VPN connection (accessing INTRANET from outside) impersonated as a trusted employee with appropriate credentials (that were stolen or extracted during snooping or eavesdropping).

b. Once data on a server is not properly secured, or data access is not authenticated from within INTRANET, an Intruder can easily breach security or bypass access rights (e.g. weak, well known or no password) or access data directly when hooked up to intranet with his own machine.

Following assumptions (for example #1) were taken:

- there might be basic (based on password type "something I know") or advanced authentication smart card / token
- TCP/IP system access with MAC address verification
- vital company data is stored on server
- credentials can be stolen or snooped
- MAC address can be altered

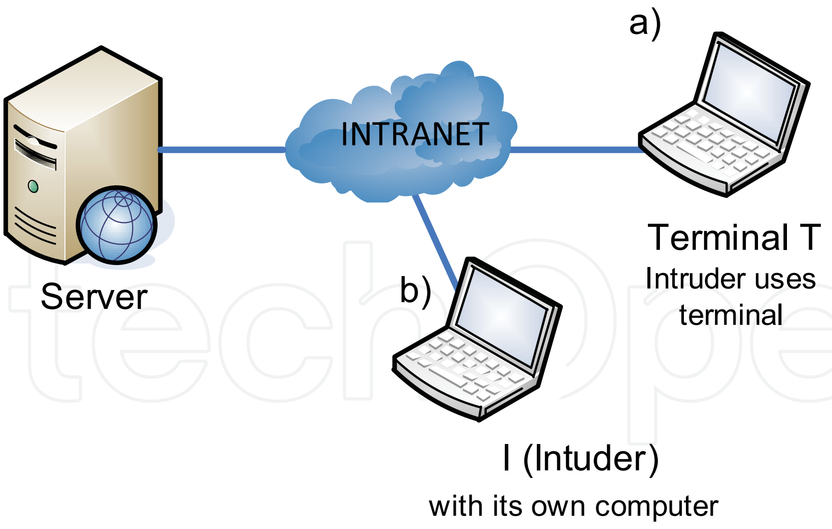


Fig. 1. (Example #1) Scenario – "The Intruder has access to the data stored on server"

2.2 Example #2: "The Victim trusts the Intruder"

This attack was already quite thoroughly described in [21]. Analyzed system is illustrated by Figure 2.

The "The Victim trusts the Intruder" attack is based on a SYN-Flood attack and goes as follows:

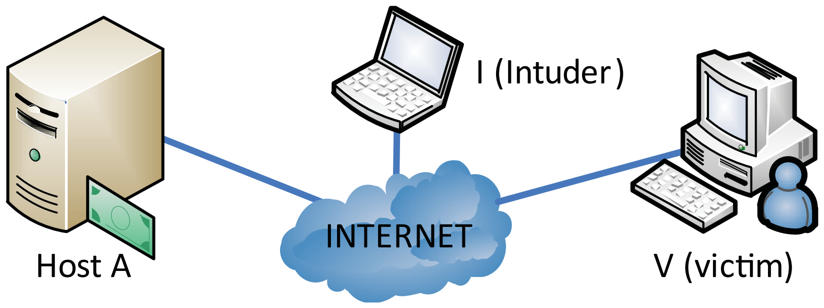


Fig. 2. (Example #2) Scenario – "Intruder-Victim-Host A"

I (Intruder) wants to pretend itself as A when talking to V. In response to the SYN packet that presumably came from A (in fact it came from I), V will reply with the ACK packet to start up a connection with A. When A sees the ACK to a request not generated by itself, it will send the RESET packet, and V drops the connection with I. In this case, the attack would fail. This situation is depicted in Figure 3.

Steps of scenarios (depicted in Figure 3) go as follows:

- Intruder sends SYN packet to V signed as Host A (a,b),
- V sends ACK packet to Host A (c),
- Host A receives ACK packet (although it has not yet sent SYN package) (d),
- Host sends RESET packet and finally connection is not being established (e, f, g).

However, let us suppose that the attacker I uses SYN-Flood. In this case, I and its co-operators send large number of SYN packets to A. This will keep A so busy, that the ACK packet sent from V is dropped and will go unnoticed by A. As a result, A will not send the RESET packet, and I can spoof A (see Fig. 4.).

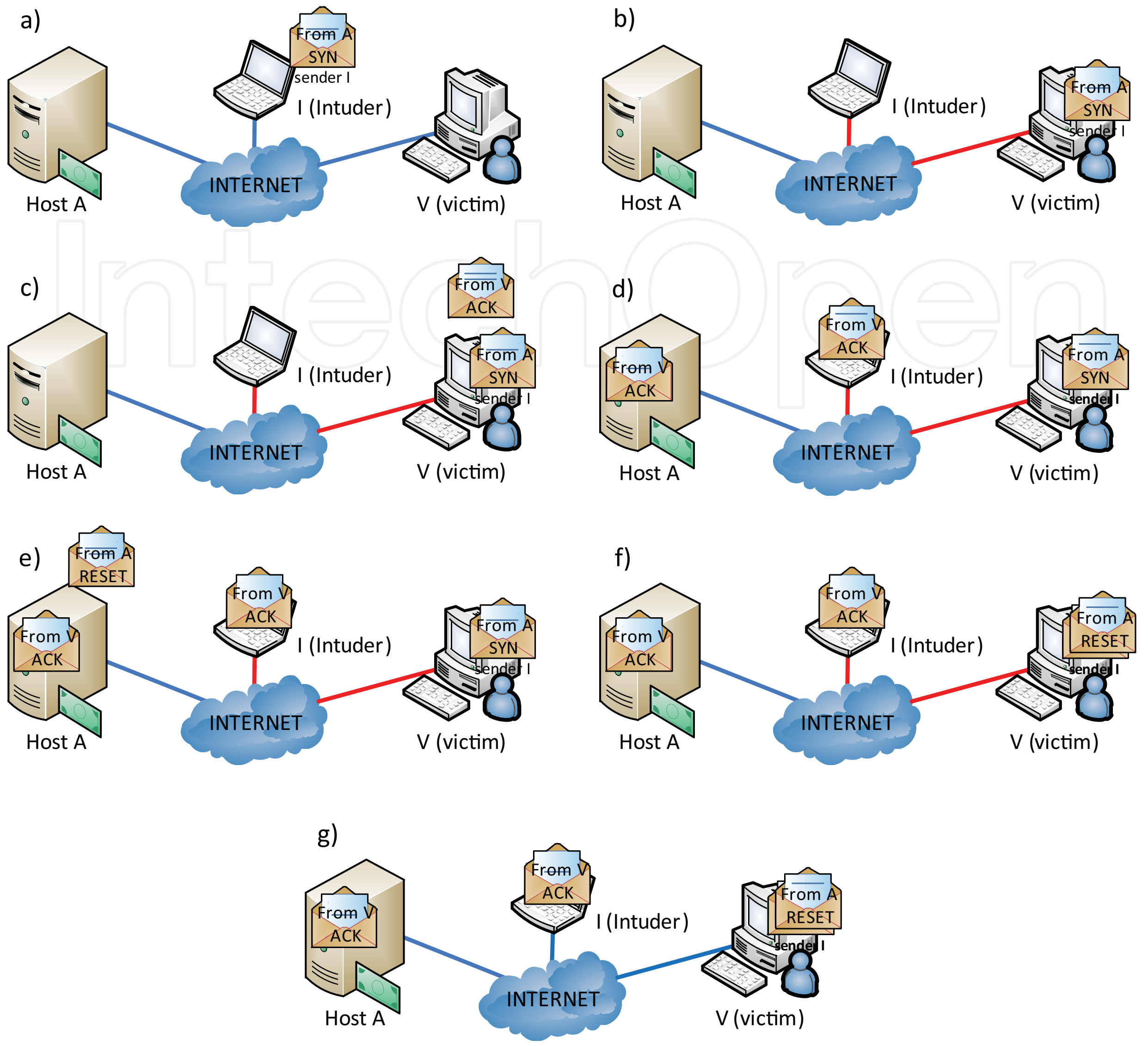


Fig. 3. “The Victim trusts the Intruder” scenario - unsuccessful attack

Steps of scenarios (depicted in Figure 4) go as follows:

- Intruder starts flooding with message Host A - DoS attack attempt
- Denial of Service attack is successful (a),
- V sends ACK packet to Host A (b),
- ACK packet is being lost (or delivery and then answer to this packet is greatly delayed)
- Connection with Intruder is established (c).

The assumptions, more detailed system description and analysis is given in [21].

3. Fault trees with time dependencies

Fault tree with time dependencies (FTTD) technique is a top-down approach. It starts from identifying all hazards (dangerous situations) in a system and their duration time. Subsequently, for each hazard, a set of such events with time parameters that can cause the hazard is generated. The steps of the construction of the FTTD and its analysis are given in Fig.5.

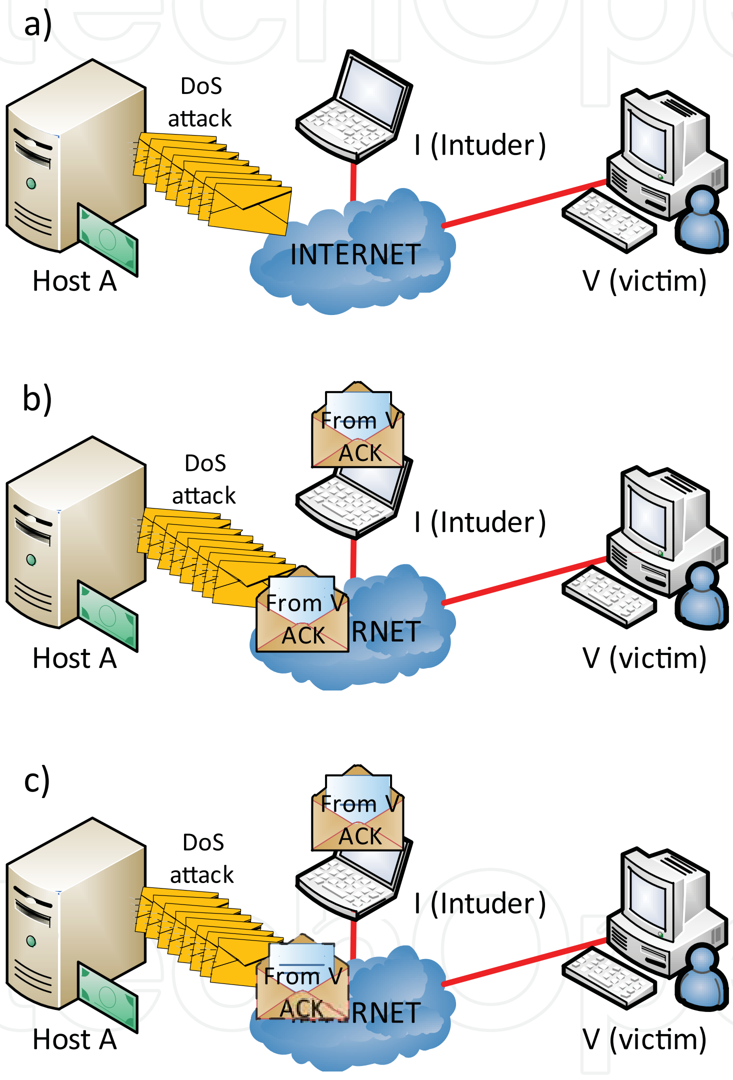


Fig. 4. "The Victim trusts the Intruder" attack based on Syn-Flood

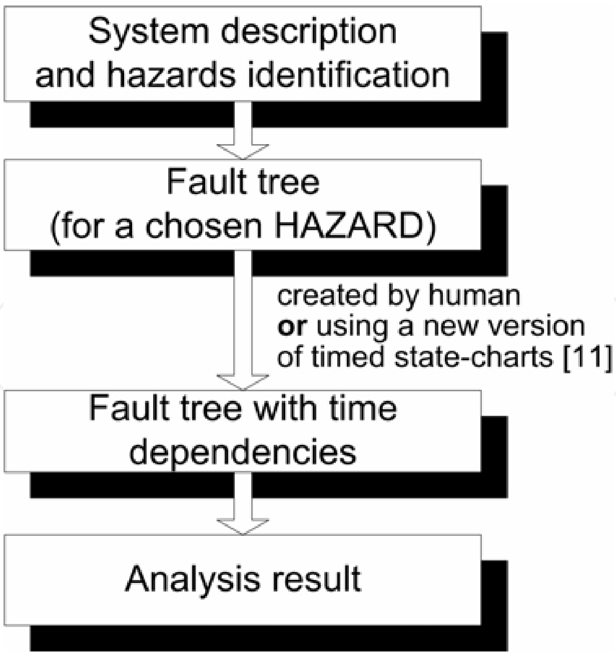


Fig. 5. The steps of the FTTD analysis

If the hazard cannot occur, or the hazard occurrence probability is on an accepted level, or the product of the hazard occurrence probability by the hazard cost is on accepted level, then we finish the analysis. Otherwise, additional security support, e.g., monitoring and security system is indispensable, as it will be shown.

3.1 The notation of the FTTD

The notations of basic gates and events are presented in Fig.6. The tool for FTTD with basic gates analysis has been presented in paper [19]. The library, as a pattern for Visio application, that allows us to draw the FTTD and to save it in file that can be analysis with tool [18]. Time dependencies can be given numerically or parametrically.

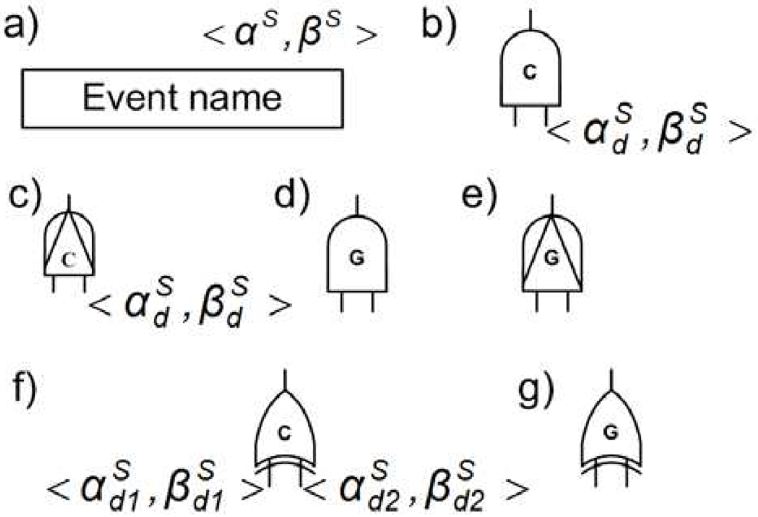

a)
$<\alpha^S, \beta^S>$
Event name
b)
C
$<\alpha_d^S, \beta_d^S>$
c)
C
$<\alpha_d^S, \beta_d^S>$
d)
G
e)
G
f)
C
$<\alpha_{d1}^S, \beta_{d1}^S>$
$<\alpha_{d2}^S, \beta_{d2}^S>$
g)
G

Fig. 6. The notations for a) events, c) causal AND gates, d) causal priority AND gates,
d) generalization AND gates, e) generalization priority AND gates, f) causal XOR gates,
g) generalization XOR gates

In the generalization gates output event is the same as one of the input events (for XOR gate) or occur when the input events coexist (for AND gate) – start corresponds to the start of event which has occurred later, and the end corresponds to end of the event which has ended earlier.

In causal gates the output event is the result of event (XOR gate) or coexistent events (AND gate) and can occur with delay to start of the cause (XOR gate) or the causes (AND gate). Duration time of the event does not depend on the cause.

An event can be an input of the gate (cause) or be an output of the gate (effect).

The meaning of symbols is as follows:

- α^S, β^S – such static parameters for the events that express the minimal and maximal duration time of the event,
- static parameters α^S, β^S for the causal AND gate are the minimal and maximal delays between the causes and the effect,
- static parameters $<\alpha^S_{d1},\beta^S_{d1}>$, $<\alpha^S_{d2},\beta^S_{d2}>$ for the causal XOR gate are the minimal and maximal delays between the two causes (1,2) and the effect; if one input is used then the parameters for second input are omitted.

The static parameters are the results of system architecture, parameters of the system components, algorithms, code execution time, and physics laws. Examples of the parameters could be the minimal and maximal times of: a transmission of a signal in the air, chemical reaction in chemical process industry, code execution, establishing the network connection, discovering the SYN-flood activity. If a duration time of an event is not known then it can be assumed that the minimal duration time is equal to 0, while the maximal one is infinity. In this case we will use the notation $<0, \infty>$. For the immediate events will use the symbol $<0,0>$.

In order to analyze the FTTD, dynamic time parameters of their events and gates have to be known. These parameters concern: the duration times of the event occurrences, delay times of the gates, and the relations among the start and end time instants of the event occurrences associated with the gates. Instead of the event occurrence, we will shortly write: event. Definitions of the FTTD gates that have been described above, will be stated below, more detailed information can be found [5], [9], [10], [13].

We will use the following notation:

- xs, xe , respectively, represents: the start, the end, respectively, of the event x,
- $\tau(xs)$ - are the time instances when the event x started,
- $\tau(xe)$ - are the time instances when the event x ended.

3.2 The causal XOR gate

Definition 1. Causal XOR gate:

$$\text{occur}(z) \Rightarrow (\text{occur } (x) \text{ and } (\ \text{duration}(x) \geq \alpha^S_{d1} \wedge \tau(xs) + \alpha^S_{d1} \leq \tau(zs) \leq \tau(xs) + \beta^S_{d1}))$$

$$\oplus\ (\text{occur } (y) \text{ and } (\text{duration}(y) \geq \alpha^S_{d2} \wedge \tau(ys) + \alpha^S_{d2} \leq \tau(zs) \leq \tau(ys) + \beta^S_{d2}))$$

where:

- α^S_{d1}, β^S_{d1} - respectively, the minimal and the maximal delays times between the occurrence of the cause x and the effect z
- α^S_{d2}, β^S_{d2} – respectively, the minimal and the maximal delays time between the occurrence of the cause y and the effect z,
- occur(z) – the predicate, the event z come into being in time, analogically: occur(x), occur(y),

- duration(x) – the predicate, event *x* duration in time, analogically: duration(y),
- ⊕, ∧ – the logical symbol denotes respectively, "exclusive disjunction", "logical conjunction".

The models of the causal XOR gates can be generalized by the causal XOR gates with: more than two input events and for one input only.

The Causal XOR gate with output event z and two inputs events x, y is given in Fig.7.

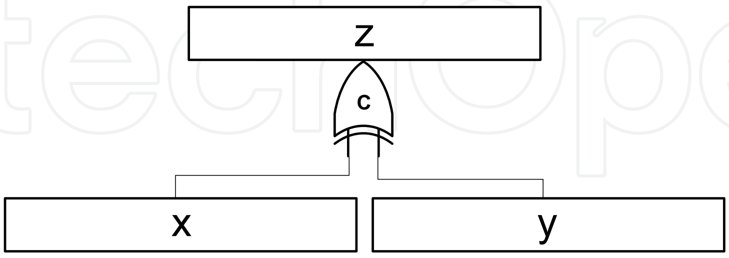


Fig. 7. The Causal XOR gate

The time relations between event *x* (cause) end event *z* (effect) are given in Fig. 8.

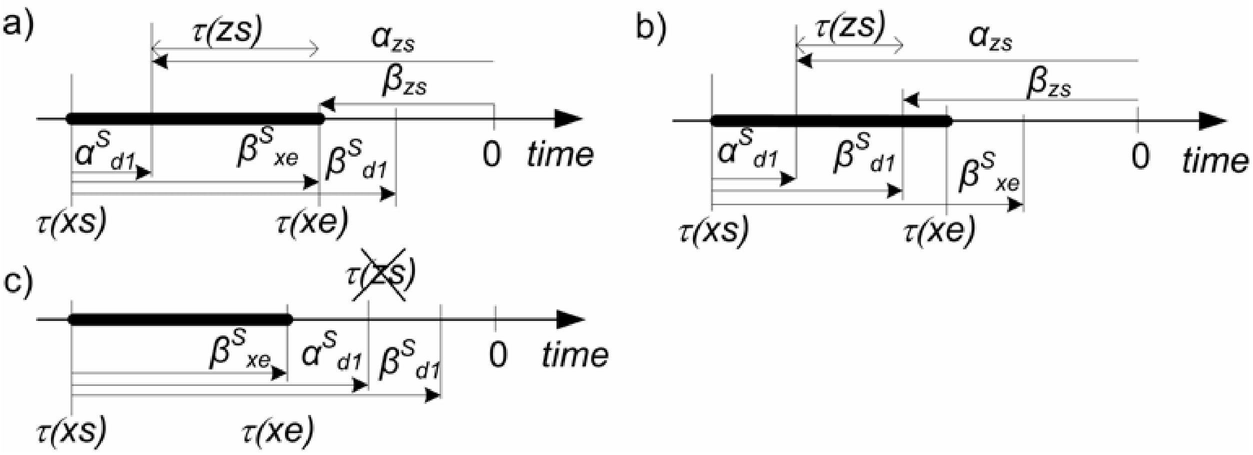


Fig. 8. The time relations between event *x* (cause) and event *z* (effect) for causal XOR gate

In causal XOR Gate the effect (event *z*) must occur not earlier that $\alpha^{S}{}_{d1}$ and not later that $\beta^{S}{}_{d2}$ counting from the start of causes (event *x*). Additionally, the effect can occur only before the end of cause (before τ(xe)). There are some examples (presented in Fig. 5) of time intervals in which event *z* can occur:

- start of the event *z* must occur between: τ(xs)+ $\alpha^{S}{}_{d1}$ and τ(xs)+ $\beta^{S}{}_{xe}$ (see Fig. 5a), because maximal duration of the event *x* is smaller than maximal delay time between causes and effect ($\beta^{S}{}_{xe}\leq \beta^{S}{}_{d1}$),
- start of the event *z* must occur between: τ(xs)+ $\alpha^{S}{}_{d1}$ and τ(xs)+ $\beta^{S}{}_{d1}$ (see Fig. 5b), because maximal delay time between causes and effect is smaller than maximal duration of the event *x* ($\beta^{S}{}_{xe}\leq \beta^{S}{}_{d1}$),
- event *z* cannot occur, because duration of the event *x* is too small (see Fig. 5c).

Let us assume minimal and maximal time for start and end of the event *z* are respectively α_{zs}, β_{zs}, therefore $\alpha_{zs}\leq\tau(xs)\leq\beta_{zs}$. Knowing that τ(xs)+ $\alpha^{S}{}_{d1}\leq$ τ(zs) and τ(zs)≤ τ(xs)+ min{ $\beta^{S}{}_{d1}$, $\beta^{S}{}_{xe}$} hence α_{zs}= τ(xs)+ $\alpha^{S}{}_{d1}$ and β_{zs}= τ(xs)+ min{ $\beta^{S}{}_{d1}$, $\beta^{S}{}_{xe}$} – we have knowledge about time relation between causes and effect.

On the other hand, if α $_{zs}$, β_{zs} are known then we can calculate minimal and maximal time in which event *x* can occur – what it can cause effect *z* (we can calculate α $_{xs}$, β_{xs}). This knowledge allows us for elaboration of the inequalities – equalities system for each gate. Derivations of those formulas are of no requirement for our discussion (more information are given in [10], [13]), only the basic knowledge of the gate model and final form of its inequalities-equalities system is required. For causal XOR gate the inequalities-equalities system is given by formula (1).

It is possible to use fault tree with time dependencies tools [18] as well. Using such tools it is essential for the construction the fault tree to use the toolbox in MS Visio or to prepare a file as it was shown in [19].

$$\begin{cases} a_{d1}^S \le \beta_{xe}^S \\ a_{xs} = a_{zs} - min\{\beta_{d1}^S, \beta_{xe}^S\}, \quad \beta_{xs} = \beta_{zs} - a_{d1}^S \\ a_{xe} = a_{zs}, \beta_{xe} = \beta_{xs} + \beta_{xe}^S \end{cases} \tag{1a}$$

$$\oplus \begin{cases} a_{d2}^S \le \beta_{ye}^S \\ a_{ys} = a_{zs} - min\{\beta_{d2}^S, \beta_{ye}^S\}, \ \beta_{ys} = \beta_{zs} - a_{d2}^s \\ a_{ye} = a_{zs}, \ \beta_{ye} = \beta_{ys} + \beta_{ye}^S \end{cases} \tag{1b}$$

(1)

3.3 The generalization AND gate

Definition 2. Generalization AND gate:

occur(z) ⇒ occur(x) ∧ occur(y) ∧ overlap(x, y) ∧
max(τ (xs), τ (ys)) = τ (zs) ∧ min(τ (xe), τ (ye)) = τ (ze)

where:

- overlap(x,y) – the predicate, the events x and y overlap in time.

The examples of time relations between events *x* and y (causes) end event *z* (effect) are given in Fig. 9.

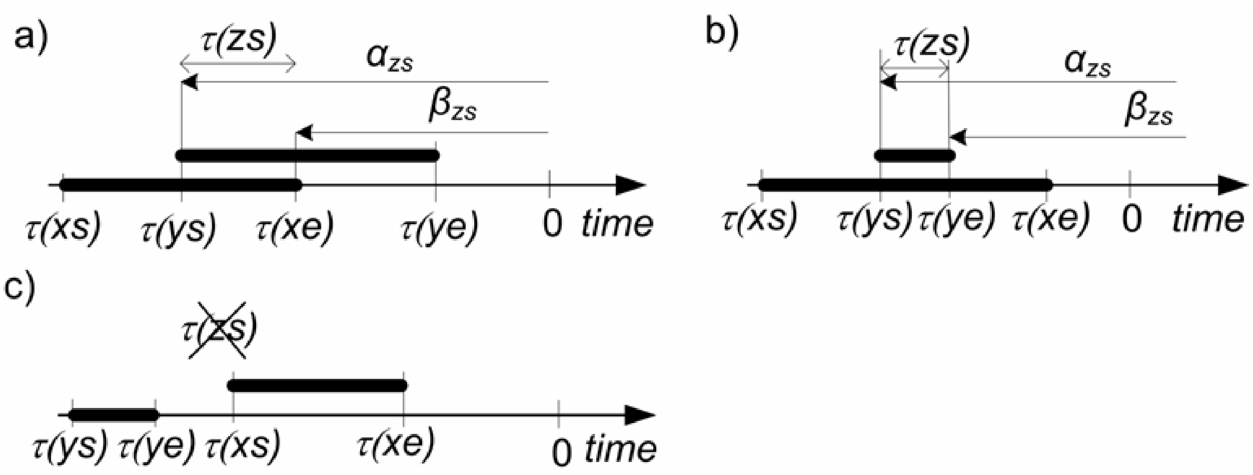


Fig. 9. The time relations between event *x* (cause) and event *z* (effect) for generalization AND gate

The inequalities-equalities system is given by formula (2).

$$
\begin{aligned}
&\begin{cases} a_{xs} = a_{zs}, \beta_{xs} = \beta_{zs}, a_{xe} = a_{ye}, \beta_{xe} = \beta_{xs} + \beta_{xe}^{S} \\ a_{ys} = a_{xs} - \beta_{ye}^{S}, \beta_{ys} = \beta_{xs}, a_{ye} = a_{ze}, \beta_{ye} = \beta_{ze} \end{cases} && \text{(2a)} \\
\oplus &\begin{cases} a_{ye}^{S} \le \beta_{xe}^{S} \\ a_{ys} = a_{zs}, \beta_{ys} = \beta_{zs}, a_{ye} = a_{ze}, \beta_{ye} = \beta_{ze} \\ a_{xs} = a_{ye} - \beta_{xe}^{S}, \beta_{xs} = \beta_{ys}, a_{xe} = a_{ye}, \beta_{xe} = \beta_{xs} + \beta_{xe}^{S} \end{cases} && \text{(2b)} \\
\oplus &\begin{cases} a_{ys} = a_{zs}, \beta_{ys} = \beta_{zs}, a_{ye} = a_{xe}, \beta_{ye} = \beta_{ys} + \beta_{ye}^{S} \\ a_{xs} = a_{ys} - \beta_{xe}^{S}, \beta_{xs} = \beta_{ys}, a_{xe} = a_{ze}, \beta_{xe} = \beta_{ze} \end{cases} && \text{(2c)} \\
\oplus &\begin{cases} a_{xe}^{S} \le \beta_{ye}^{S} \\ a_{xs} = a_{zs}, \beta_{xs} = \beta_{zs}, a_{xe} = a_{ze}, \beta_{xe} = \beta_{ze} \\ a_{ys} = a_{xe} - \beta_{ye}^{S}, \beta_{ys} = \beta_{xs}, a_{ye} = a_{xe}, \beta_{ye} = \beta_{ys} + \beta_{ye}^{S} \end{cases} && \text{(2d)}
\end{aligned}
\tag{2}
$$

3.4 The generalization XOR gate

Definition 3. Generalization XOR gate:

$occur(z) \Rightarrow (occur(x) \wedge x = z) \oplus (occur(y) \wedge y = z)$

The example of time relations between events x (cause) and event z (effect) are given in Fig. 10.

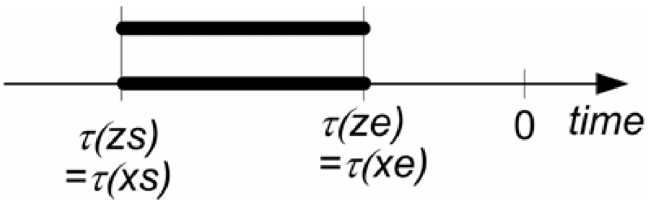


Fig. 10. The time relations between event x (cause) and event z (effect) for generalization XOR gate

The inequalities-equalities system is given by formula (3).

$$
\begin{aligned}
&\begin{cases} a_{xs} = a_{zs},\ \beta_{xs} = \beta_{zs} \\ a_{xe} = a_{ze},\ \beta_{xe} = \beta_{ze} \end{cases} && \text{(3a)} \\
\oplus &\begin{cases} a_{ys} = a_{zs},\ \beta_{ys} = \beta_{zs} \\ a_{ye} = a_{ze},\ \beta_{ye} = \beta_{ze} \end{cases} && \text{(3b)}
\end{aligned}
\tag{3}
$$

The other gates are not presented here as they do not occur in the FTTDs for given examples.

The inequalities – equalities systems for other gates can be found in papers [9], [10], [13]. These systems are used in backward analysis from the hazard event to primary events.

4. FTTD analysis

In order to simplify the interpretation of the result analysis, the conventional moment of time „0" has to be established as start of the hazard, therefore:

$$\alpha_{hs} = 0 \text{ and } \beta_{hs} = 0 \tag{4}$$

We start the FTTD analysis by identifying the time interval, when the hazard can occur using formula (4). More information about it is given in [10].

$$\alpha_{he} = \alpha^{S}{}_{he} \text{ and } \beta_{he} = \alpha^{S}{}_{he} \tag{5}$$

Then, we consider the causes that can directly lead to the hazard and time relations between these causes and the hazard occurrences using formulas for adequate gates (for each hazard is the output event). Then we consider causes for the above causes, etc.

As a result of this analysis, the following pairs for the events are obtained:

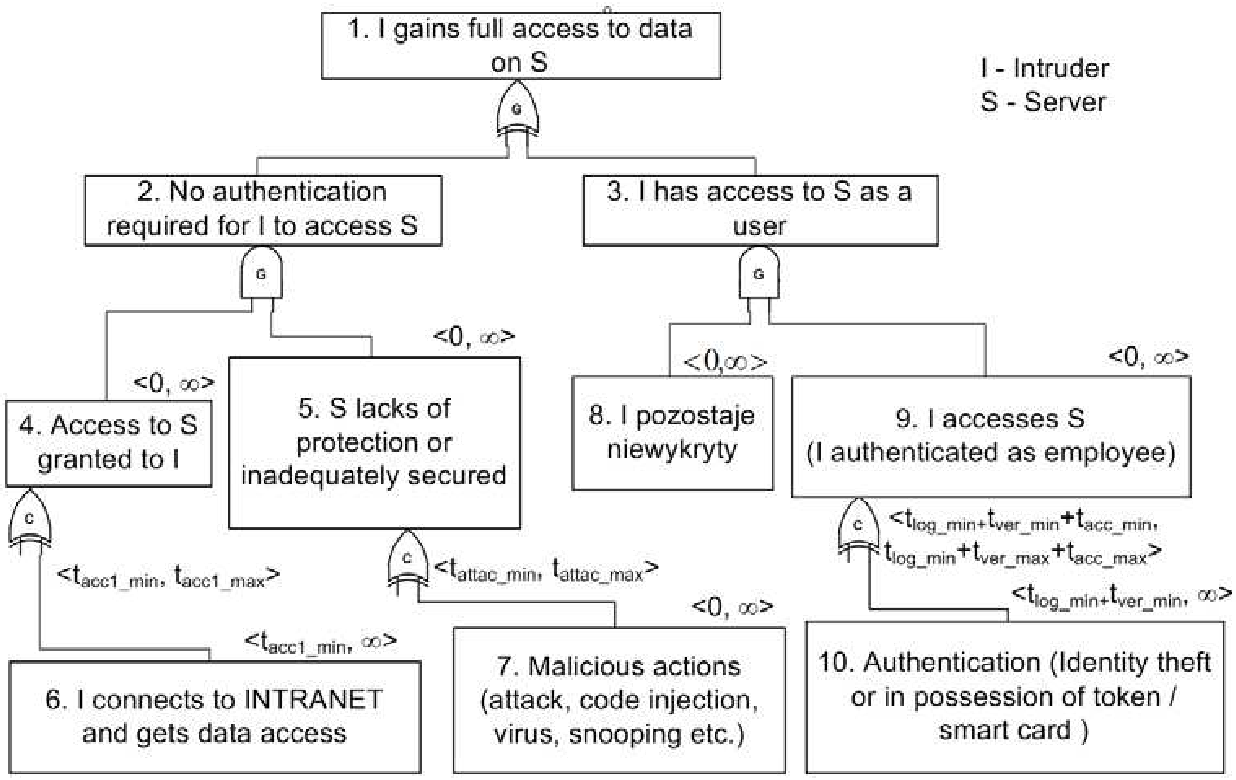
$<\alpha_{zs}, \beta_{zs}>$, $<\alpha_{ze}, \beta_{ze}>$ where:

- α_{zs}, β_{zs}, respectively, are be the earliest, the latest time instant of the start of the event z,
- α_{ze}, β_{ze}, respectively, be the earliest, the latest time instant of the end of the event z.

These are time constraints imposed on events occurrence time in order to cause the hazard.

4.1 FTTD analysis - example #1

The fault tree for the system (called example #1 - depicted on Fig. 1.) can be found at Fig. 11. The left sub-tree (composed of events: 2, 4, 5, 6, 7) corresponds to the scenario (a) represented by Fig. 1 - the intruder uses the terminal T impersonated as the worker to get



1. I gains full access to data on S
I - Intruder
S - Server
G
2. No authentication required for I to access S
3. I has access to S as a user
G
G
<0, ∞>
4. Access to S granted to I
<0, ∞>
5. S lacks of protection or inadequately secured
<0,∞>
8. I pozostaje niewykryty
<0, ∞>
9. I accesses S (I authenticated as employee)
C
<t_acc1_min, t_acc1_max>
C
<t_attac_min, t_attac_max>
C
<t_log_min+t_ver_min+t_acc_min, t_log_min+t_ver_max+t_acc_max>
<t_log_min+t_ver_min, ∞>
<0, ∞>
<t_acc1_min, ∞>
6. I connects to INTRANET and gets data access
7. Malicious actions (attack, code injection, virus, snooping etc.)
10. Authentication (Identity theft or in possession of token / smart card)

Fig. 11. The FTTD for Example #1

access to data. The right sub-tree (composed events 3, 8, 9, 10) corresponds to the scenario (b) in which Intruder uses his own laptop to get direct access to compromised (unsecured) server.

Following steps should be undertaken in order to get meaningful result after the FTTD analysis.

STEP 1: The calculations of the time parameters for HAZARD (event 1) using formulas (4) and (5) are being done. As hazard is the output event in a generalization gate, the static parameters for hazard must be obtained from the input event. If they are equal, then we have one case (as in our case), if they are not - we must consider more cases as it was shown in [13].

Hence: **1 <0,0> <0, ∞>**, where: 1 – number of top event, $<\alpha_{1s}=0, \beta_{1s}=0> < \alpha_{1e}=0, \beta_{1e}=\infty>$. The event 1 starts in time instant "0" ($\alpha_{1s} \leq \tau(1s) \leq \beta_{1s,}$ then $0 \leq \tau(1s) \leq 0$ and $\tau(1s)=0$), the event 1 end instant satisfies the inequalities $0 \leq \tau(xe) \leq \infty$.

STEP 2: The calculations for the events 2 and 3.

We make the calculation from formulas (3) for the generalization XOR gate.

We have: 2 <0,0> <0, ∞>
  xor 3 <0,0> <0, ∞>

STEP 3: The calculations for the events 4 and 5.

We make the calculation from formulas (2) for the generalization of AND gate and for time parameters for the event 2: $<\alpha_{2s}=0, \beta_{2s}=0> < \alpha_{2e}=0, \beta_{2e}=\infty>$.

We have: (4 <0,0> <0, ∞> and 5 <-∞,0> <0, ∞>)
  xor (4 <-∞,0> <0, ∞>) and 5<0,0> <0, ∞>)

We have two cases (the following two are the equal):

- the event 5 occurs before the event 4 (then event 5 start must satisfy the inequalities: $-\infty \leq \tau(5s) \leq 0$) and the event 5 must end not earlier than event 4 starts (the event 5 end instant must satisfy the inequalities: $0 \leq \tau(5e) \leq \infty$); the event 4 start is in time instant "0", the event 4 end satisfies the inequalities $0 \leq \tau(4e) \leq \infty$,
- the event 4 occurs before the event 5 and the event 4 must end not earlier than the event 5 starts: ($-\infty \leq \tau(2s) \leq 0$, $0 \leq \tau(2e) \leq \infty$, $0 \leq \tau(3s) \leq 0$, $0 \leq \tau(3e) \leq \infty$).

STEP 4: The calculation for the event 8 and 9 – are analogous as for 4 and 5 using formulas(2).

NEXT STEPS: We analyze remaining events using formula (1b). The result of the analysis is depicted on Fig. 12.

Each rectangle at Fig. 12 has the Minimal Cut Set (MCS). According to the paper [3] (excluding inscription: *or in a proper time*): Minimal cut set (MCS) - is such minimal set of events that if they all occur simultaneously (or in a proper sequence *or in a proper time*) then the top event occurs. If one event from MCS does not occur, then it causes that top event will also not occur.

Words in bracket:

- „in a proper sequence" - extends standard definition of MCS of FTA to events occurring sequence dependencies,
- „*in a proper time*" - extends standard definition of MCS of FTA to time dependencies, what is required for the FTTD.

MCS for FTTD includes events with time conditions. For the event i we have:

$$i <\alpha_{is}, \beta_{is} > < \alpha_{ie}, \beta_{ie}>$$

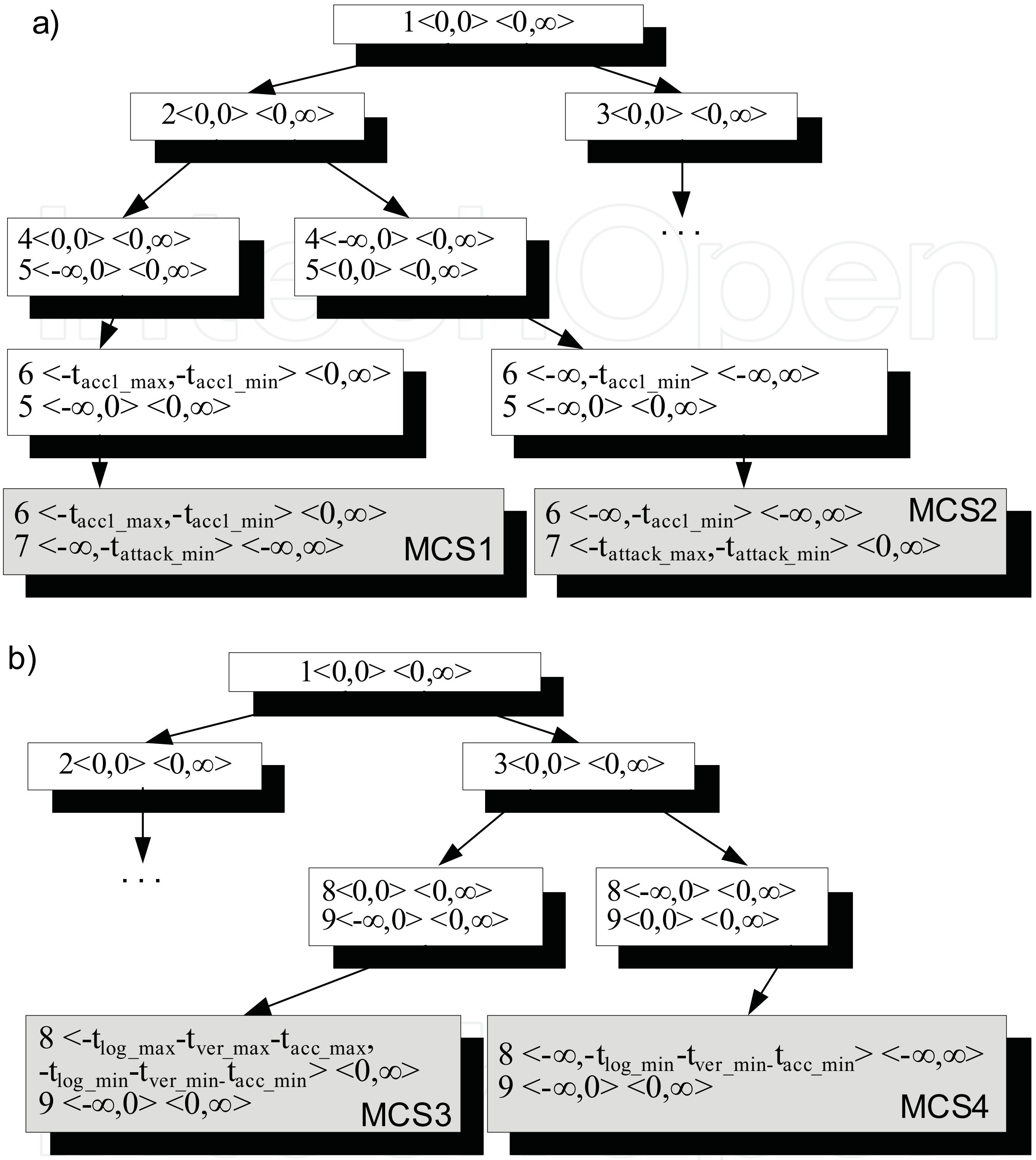

a)
1<0,0> <0,∞>
2<0,0> <0,∞>
3<0,0> <0,∞>
. . .
4<0,0> <0,∞>
5<-∞,0> <0,∞>
4<-∞,0> <0,∞>
5<0,0> <0,∞>
$6 <-t_{acc1\_max},-t_{acc1\_min}> <0,\infty>$
$5 <-\infty,0> <0,\infty>$
$6 <-\infty,-t_{acc1\_min}> <-\infty,\infty>$
$5 <-\infty,0> <0,\infty>$
$6 <-t_{acc1\_max},-t_{acc1\_min}> <0,\infty>$
$7 <-\infty,-t_{attack\_min}> <-\infty,\infty>$
MCS1
$6 <-\infty,-t_{acc1\_min}> <-\infty,\infty>$
MCS2
$7 <-t_{attack\_max},-t_{attack\_min}> <0,\infty>$
b)
1<0,0> <0,∞>
2<0,0> <0,∞>
3<0,0> <0,∞>
. . .
8<0,0> <0,∞>
9<-∞,0> <0,∞>
8<-∞,0> <0,∞>
9<0,0> <0,∞>
$8 <-t_{log\_max}-t_{ver\_max}-t_{acc\_max},$
$-t_{log\_min}-t_{ver\_min-}t_{acc\_min}> <0,\infty>$
$9 <-\infty,0> <0,\infty>$
MCS3
$8 <-\infty,-t_{log\_min}-t_{ver\_min-}t_{acc\_min}> <-\infty,\infty>$
$9 <-\infty,0> <0,\infty>$
MCS4

Fig. 12. FT analysis for Example #1 a) left sub-tree, b) right sub-tree

The time conditions denotes, respectively: the earliest (α_{is}) and the latest (β_{is}) time instants of the event i start, and the earliest (α_{ie}) and the latest (β_{ie}) time instants of the event i end. In order to cause the hazard (top event), the event i start instant ($\tau(is)$) and the event i end instant ($\tau(ie)$) must satisfy the inequalities: $\alpha_{is} \leq \tau(is) \leq \beta_{is}$, $\alpha_{ie} \leq \tau(ie) \leq \beta_{ie}$.

The α_{is}, β_{is}, α_{ie}, β_{ie} can be counted with respect to reference time instant 0 (e.g. we assume $\beta_{1s}=0$, see [10]).

If one event from MCS will not occur in the proper time (e.g. as result of introduced insurance, catalysts, equipment with proper parameters), it causes that the top event will also not occur.

Finally, the sets marked with gray color have been obtained. The final MCS have only events which are leaves of the FTTD.

Interpretation for the MCS1 and MCS2:

- Events 6 and 7 must occur together to cause the effect, we known that the duration time before effect must be minimally equal to t_{acc1_min} (from MCS1), t_{attack_min} (from MCS2), therefore

$$\min\{t_{acc1\_min}, t_{attack\_min}\}$$

Interpretation for the MCS2 and MCS3:

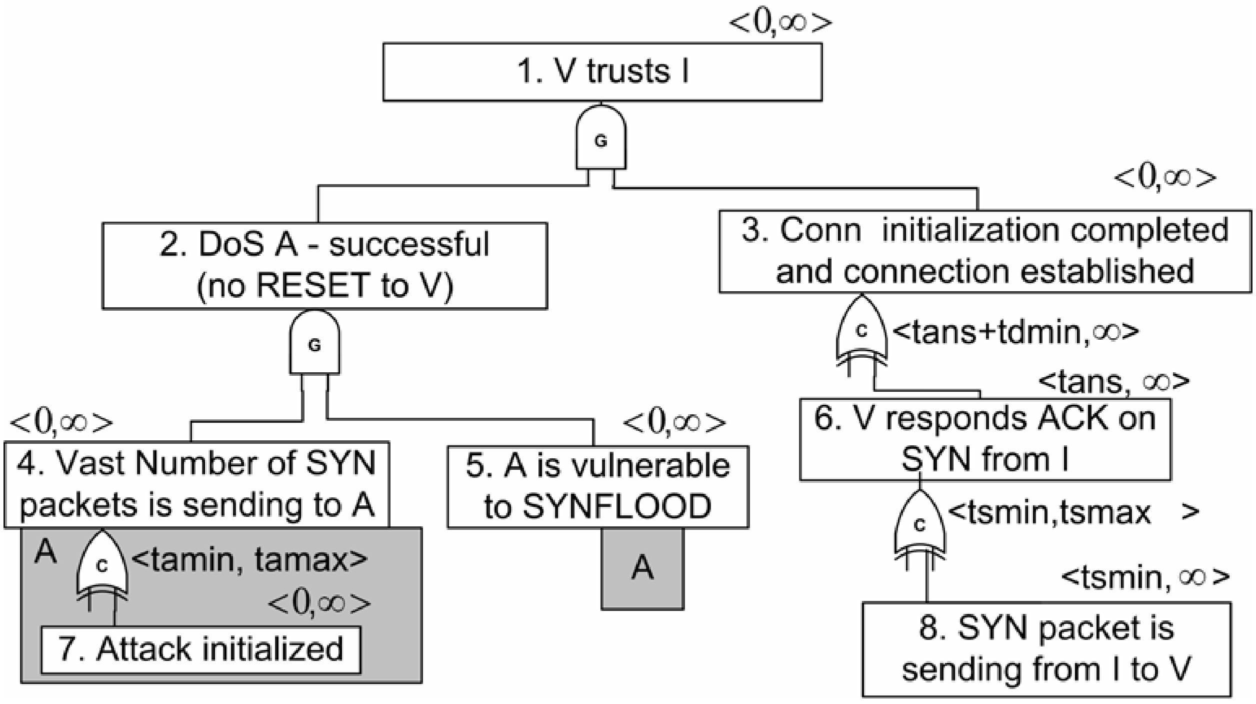
- analogically as it was for the events 6 and 7, it has minimal hazard time equal to

$$t_{log\_min}+t_{ver\_min}+t_{acc\_min}.$$

The use of FTTD is this case is not required; however taking advantage of formal methods is beneficial; it prevents creation an unequivocal description, we are able to make the precise specification and allow avoiding omission not so obvious aspects.

4.2 FTTD analysis - example #2

The fault tree for the system (example #2 - depicted on Fig. 2.) can be found at Fig. 13 (for more detailed information please refer to [21]).



<0,∞>
1. V trusts I
G
<0,∞>
2. DoS A - successful (no RESET to V)
3. Conn initialization completed and connection established
C
<tans+tdmin,∞>
G
<tans, ∞>
<0,∞>
<0,∞>
6. V responds ACK on SYN from I
4. Vast Number of SYN packets is sending to A
5. A is vulnerable to SYNFLOOD
C
<tsmin,tsmax >
A
C
<tamin, tamax>
A
<0,∞>
<tsmin, ∞>
7. Attack initialized
8. SYN packet is sending from I to V

Fig. 13. The FTTD for "V trusts I" attack

As it was presented in [21], the time parameters for the XOR causal gates of the right part of the FTTD are as follows:

- tsmin, tsmax - the minimal and maximal time of sending the SYN packet from the I to V,
- tans – time of preparing the acknowledgement ACK on the SYN packet, it is the activity of V,

- tdmin – minimal time of establishing the connection between I and V after time instant when V have sent the ACK.

Event "Attack initialised" means the sending of the first SYN packet from V to A, i.e., SYN-Flood has been started.

The time parameters for the XOR causal gate of the left part of the FTTD are as follows:

- tamin, tamax – minimal and maximal lengths of time interval which is sufficient to send such a number of SYN packets from I to A that A cannot generate the RESET packet.

The event "A is vulnerable to SYN-Flood" means that there are no such facilities at A that A can avoid the SYN-Flood.

If the events 4. and 5. are overlapped in time then the Denial of Service occurs and A cannot send the RESET packet to V ("DoS A – successful (no RESET to V)").

Let the events 2. and 3. are overlapped in time. Hence, the connection between I and V has been established, and A is not able to send the RESET packet to V. Therefore, "V trusts I".

The more detailed analysis was shown in [21]. Result of the analysis is depicted on Fig. 8.

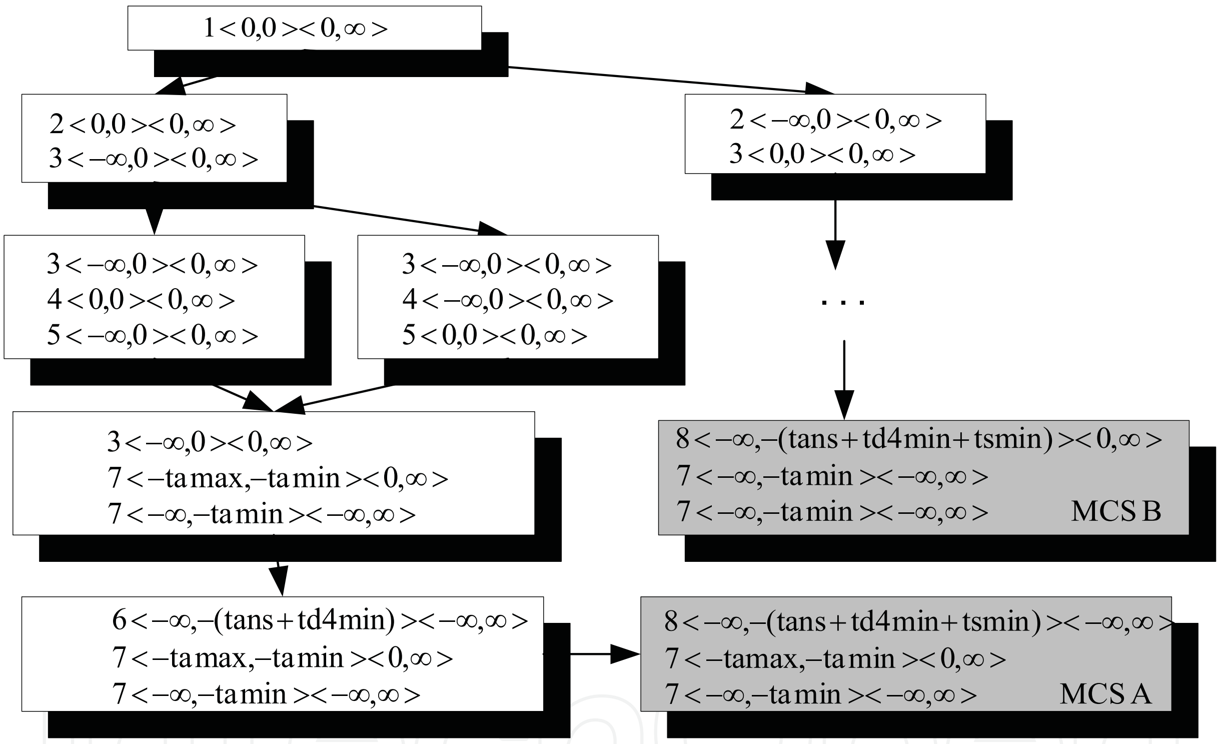


Fig. 14. The analysis result

Interpretation for the MCS A (please refer to [21]):

1. time intervals connected with the event 7 are different; so after calculating the common part, we have: 7 <-tamax,-tamin> <0,∞> and 8 <-∞, -(tans+tdmin+tsmin)> <-∞,∞>,
2. counting from sending the SYN package (the event 8), the hazard "V trusts I" will occur minimally after tans+tdmin+tsmin time units, but the attack needs to last until the event "V trusts I".

Interpretation for the MCS B:

The hazard can occur, when the event "Attack initialized" starts not later than tamin before the hazard. Analogically the event 8 starts not later than tans+tdmin+tsmin and ends – not earlier than in 0 time instant.

5. Application of result received in analysis

5.1 System with protection

The abovementioned timing properties can be used in designing protections and services. Now three proposals will be presented.

1. If the time connected with service activity of one SYN packet is t_{SYN}, then the amount of packets that can be serviced by server in time before "V truts I" is about (tans+tdmin+tsmin)/t_{SYN}. Hence, these packet numbers can be used for construction of the firewall rules or for specification of the requirements of attack detection,
2. if in order to avoid the vulnerability to SYN-Flood attack, a protection, e.g. SYN Cookies, should be introduced or reintroduced in time no longer than t_{amin}; but if the computer Intruder sends the packet (SYN to the V) simultaneously with the attack initialization, then the protection has to complete its procedure in time not longer than max {tamin, tans+tdmin+tsmin},
3. if the security is more important than functionality, then knowing the time parameters connected with the attack is essential, we can, e.g. monitor the load of the server (the testing of the SYN packets). If the server cannot respond before the time t_{ans}+t_{dmin}+t_{smin}, then the RESET packets can reset all connections,
4. Vulnerability of the server during the attack cannot be greater than min{t_{acc1_min}, t_{attack_min}}, hence we are able to periodically verify server protection,
5. Intruder must be detected in t_{log_min}+t_{ver_min}+t_{acc_min} time units; therefore e.g. "visual" verification of employees by a guard or cameras with face recognition system should be introduced.

5.2 Safety audit procedure

The efficient and safe IT assets management requires detailed knowledge of the infrastructure and underlying inventory items. All information that one may need for that purpose can be acquired during a security audit.

The security audit is a process in which collection and evaluation of "evidence and premises" is taking place in order to determine whether the computer system and related resources are properly secured; to retain data integrity and desired level of security, and to provide relevant and reliable information that allow quickly and effectively achieve organization security objectives. Audit results can influence on resources utilization (to be used sparingly), enforce internal control mechanisms to provide reasonable means to assure that operational objectives be achieved and help control IT environment by enabling protection against adverse events or ultimately help in detection on time to minimize the effects that might have been already caused.

The fault trees can be used to support construction of security audit procedure. The common knowledge is that any system as a whole is as weak as its weakest component, so each protection is equally important. Having this in mind, along with fault tree constructed and then analyzed one can be tempted to create a security survey that will constitute a basis for fully-fledged security audit. Conclusions drawn after the analysis can be formulated as questions for administrators toward the systems that are about to be secured.

Let us focus on two examples based on aforementioned assumptions:

1. Having considered Fault Trees without time dependencies, one knows that events 4, 5 and 8 are only prerequisites for hazard to occur. Based on that following questions may be added to a security survey:

- Is system still synflood attack prone?
- Has synflood susceptibility been eliminated or considerably restricted?
- Had SYN packets number being received from one source been limited?
- If in a corporate environment, has any additional authentication (client's terminal and server) mechanism been introduced?

2. Having analyzed FT and a monitoring tool in place, one may want to create a metric that will be delivering information about a number of incoming packets in a defined timeframe.

These are just simple examples that could be extended in a wide variety of cases to enforce security awareness and apply security rules. What's more based on the response from the security survey generic firewall rules may be created and instantly adapted e.g.

If system Y attack X prone then block INCOMING communication using protocol XYZ on port xxx.

Having in mind example #1, one may be tempted to draw some conclusions out of its analysis.

1. Any person that is logged in should be double checked if he/she is the really the one that has been logged as. So following security rule can be established

If a user is successfully logged, an additional identity check should be performed not later than … since the time when authentication was successful.

2. Automated security check ups should be employed as a part of periodical audit routines (penetration tests, scripting based logging procedures etc.)

6. Conclusions

In the chapter, the FTTD analysis has been applied in pursue for such minimal sets of causes with time relations that can lead to the hazard. The hazard is the event when the security requirements are violated. It has been shown that the time relations between the events and the hazard could be applied in: construction of protections against the hazard occurrence or such a choice of network parameters that the hazard can be avoided. Additionally, the fault tree can be used for identification of security threats; and its analysis may help in introduction of attacks remedial measures, like discussed example of safety audit procedure. The FTTD analysis can be used not only for analysis of faults consequences, but it can be applied for checking whether a design of a network satisfies some safety requirements, too.

What is more, there are such tools that can automate and help with the analysis of FTTD with four types of gates: generalization AND, XOR, priority AND and causal AND, XOR, priority AND.

7. References

[1] J. F. Allen, G. Ferguson, "Actions and events in interval temporal logic", Journal of Logic and Computation, Vol. 4, 1994, No. 5, pp. 531-579.

[2] D. J. Bernstein's, SYN Cookies explanation, http://cr.yp.to/syncookies.html

[3] "Fault Tree Analysis (FTA)", International Technical Commission, IEC Standard, Publication 1025, 1990.

[4] G. Helmer, J. Wong, M. Slagell, V. Honavar, L. Miller, Y. Wang, "Software Fault Tree and Colored Petri Net Based Specification, Design and Implementation of Agent-Based Intrusion Detection System", 2002.

[5] J. Górski, J. Magott, A. Wardziński, "Modelling Fault Trees Using Petri Nets", in: Proc. SAFECOMP'95, Belgirate, Italy, LNCS, Springer-Verlag, 1995.

[6] M. Y. Huang, T. M. Wicks, "A Large-scale Distributed Intrusion Detection Framework Based on Attack Strategy Analysis", Computer Networks, Vol. 31, 1999, No. 23-24, pp. 2465-2475.

[7] Introduction to TCP/IP (http://www.yale.edu/pclt/COMM/TCPIP.HTM)

[8] John Kristoff - Overview of TCP (Fundamental concepts behind TCP and how it is used to transport data between two endpoints) (http://condor.depaul.edu/~jkristof/technotes/tcp.html)

[9] J. Magott, P. Skrobanek, "A method of analysis of fault trees with time dependencies", in: Proc. SAFECOMP'2000, Rotterdam, The Netherlands, LNCS, Vol. 1943, Springer-Verlag, 2000, 176-186.

[10] J. Magott, P. Skrobanek, "Method of Time Petri Net Analysis for Analysis of Fault Trees with Time Dependencies", IEE Proceedings - Computers and Digital Techniques, 2002, Vol. 149, No. 6, pp. 257-271.

[11] J. Magott, P. Skrobanek, "Partially automatic generation of fault-trees with time dependencies", in: Proc. Dependability of Computer Systems", DepCoS '06, Szklarska Poręba, Poland, IEEE Computer Society Press, 2006, 43-50.

[12] W. Richard Stevens, Detailed tutorial on TCP/IP, (http://www.goldfish.org/books/TCPIP%20Illustrated%20Vol%201/)

[13] P.Skrobanek, "A method of analysis of fault tree with time dependencies for safety-related systems" (in Polish), Ph. D. Thesis, Technical University of Wrocław, Poland, report: PRE. 24/2005 (http://www.dbc.wroc.pl/dlibra/doccontent2?id=1142&from=&from=metadatasearch&dirids=1)

[14] L. P. Swiler, C. Phillips, "A Graph-Based System for Network-Vulnerability Analysis", in: New Security Paradigms Workshop, Charlottesville, VA, USA, 1998, pp. 71-79.

[15] The basics of Transmission Control Protocol (http://tcp.mywebcities.com/)

[16] TCP, Transmission Control Protocol (http://www.networksorcery.com/enp/protocol/tcp.htm)

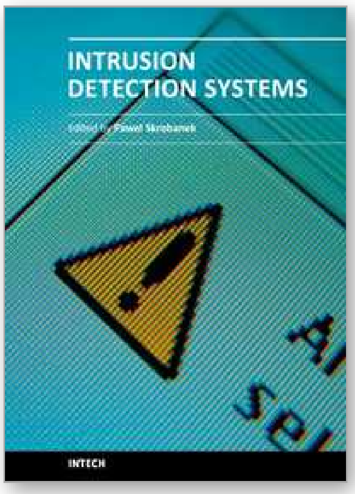
[17] G. D. Wyss, B. Schneier, T. R. Gaylor, "Probabilistic Logic Modeling of Hybrid Network Architectures", in: Proc. 21st IEEE Conference on Local Computer Networks.

[18] https://snow.iiar.pwr.wroc.pl:36914/FaultTreeWeb/

[19] M. Jureczko, P.Skrobanek, "Tool for analysis of the fault tree with time dependencies", Electrotechnical Review, 2010, R. 86, nr 9 s. 179-183

[20] Bierć P., „The safety analysis of PostgreSQL Server with PHP access using fault tree analysis", M. A. Thesis, (in Polish), Wroclaw University of Technology, Wroclaw, Poland, 2007

[21] J. Magott, P. Skrobanek, M. Woda, Analysis of timing requirements for intrusion detection systems, in: Proc. Dependability of Computer Systems, DepCoS-RELCOMEX '07, Szklarska Poręba, Poland, IEEE Computer Society Press, 2007, 278-285.



Intrusion Detection Systems
Edited by Dr. Pawel Skrobanek

ISBN 978-953-307-167-1
Hard cover, 324 pages
Publisher InTech
Published online 22, March, 2011
Published in print edition March, 2011

The current structure of the chapters reflects the key aspects discussed in the papers but the papers themselves contain more additional interesting information: examples of a practical application and results obtained for existing networks as well as results of experiments confirming efficacy of a synergistic analysis of anomaly detection and signature detection, and application of interesting solutions, such as an analysis of the anomalies of user behaviors and many others.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Pawel Skrobanek and Marek Woda (2011). Analysis of Timing Requirements for Intrusion Detection and Prevention using Fault Tree with Time Dependencies, Intrusion Detection Systems, Dr. Pawel Skrobanek (Ed.), ISBN: 978-953-307-167-1, InTech, Available from: http://www.intechopen.com/books/intrusion-detection-systems/analysis-of-timing-requirements-for-intrusion-detection-and-prevention-using-fault-tree-with-time-de

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821