Threat analysis based on the graph of elementary threats

Karel Burda
The Faculty of Electrical Engineering and Communication
Brno University of Technology, Brno, Czech Republic

Summary
At present, an analysis based on the threat tree is used for analysis of threats to information systems. Orientation to a single threat is a characteristic feature of this method. If the analysis is more extensive, then a threat tree is created for each single threat. This solution causes a complicated influence analysis of the partial threats that occur in several trees. In this paper, a method for threat analysis is described which enables a complex influence analysis of partial threats and a choice of optimal set of security countermeasures.

Key words:
threat analysis, threat tree.

1. Introduction

For an effective system security, it is necessary to know threats which can cause a loss of assets related to this system. In extensive systems, tens of threats can exist which need to be identified and analyzed.

Methods of threat analysis are used for this purpose. At present, the threat analysis, based on the so-called threat tree, is used in particular. Its origin was inspired by the method of fault analysis by means of graph tree (Fault Tree Analysis), which has been used from the early 60's of the last century for the development of systems where faults are unacceptable (e.g. nuclear reactors).

According to [1], the fault tree is a graphic model of the various parallel and sequential combinations of faults that will result in the occurrence of the predefined undesired event. The faults can be events that are associated with component hardware failures, human errors, software errors, or any other pertinent events which can lead to the undesired event. A fault tree thus depicts the logical interrelationships of basic events that lead to the undesired event, the top event of the fault tree.

The same concept was applied to the threat analysis in [2]. In this case, the undesired event is the realization of any threat and the basic events are partial threats whose combination forms the given threat. In 1993 [3], this idea was refined and popularized in the so-called attack tree method. Here, the undesired event is the realization of an attack on a given asset, and the basic events are the realizations of partial attacks whose combination leads to a successful attack on a given asset.

The characteristic feature of an analysis by means of the threat tree is the orientation to a single threat. If the system is more extensive, then several threat trees are created for each single threat [4]. This solution, however, leads to a more complex influence analysis of partial threats which occur in several trees.

In this paper, the method for threat analysis is described, which is suitable for complicated and extensive systems. In the first phase, mutually independent and disjunctive partial threats (so-called elementary threats) are defined in a systematic way. In the next phase, system threats (so-called combined threats) are defined from the elementary threats using the Boolean logic. Then we can find out the influence of single elementary threats on system threats by analyzing the logic functions obtained.

2. Graph of elementary threats

We define a threat H as a possibility of loss of assets (e.g. confidential data loss, non-availability of services, etc.). Each threat H is either divided into its partial threats H_x or it is useless to divide this threat for a given depth of description (e.g. it is useless to divide the threat of a smart card theft into its various possibilities for the purpose of analyzing the encryption key loss threat in this card). We call the threat which is not further divided an elementary threat. Dividing threats into partial threats is accomplished according to the following requirement P.

Requirement P: Threat H is systematically divided into n partial threats H_x where $x \in \{1, 2,..., n\}$ so that these partial threats are mutually disjunctive and their union represents the threat H fully.

We can formally express the requirement P as follows:

$$H = \bigcup_{x=1}^{n} H_x, \tag{1}$$

where

$$H_x \cap H_y = \varnothing \text{ for } x \neq y, \tag{2}$$

where $\varnothing$ is an empty set and indices $x, y \in \{1, 2,..., n\}$.

The criterion for dividing a threat into its partial threats is individual for each threat. It can be e.g. the criterion of threat source (attacker, user, operator, fault, natural disaster), the criterion of asset type (hardware, software, data), the criterion of security attribute (confidentiality, authenticity, availability), the criterion of attack taxonomy, etc.

Manuscript received December 5, 2008
Manuscript revised December 20, 2008

The requirement P enables creating a relatively complete list of mutually independent elementary threats whose logical combinations form the so-called combined threats. In this context, it is desirable to emphasize the word relatively. This is to say, an absolutely reliable method for threat analysis cannot exist, because neglecting a threat is an unavoidable risk of each method. The risk mentioned above is due to the problem that if any threat is unknown to the threat analyst then this does not mean that this threat does not exist or cannot exist in future. This is because new threats appear with the progress of technology and human knowledge. The following algorithm is used to generate a list of elementary threats.

Algorithm for generating an oriented graph G of threats:

1. Subgraph G' consists of a single node U_0, which represents the highest threat H_0, i.e. the threat against security of the system.
2. In subgraph G' we find leaf U_i, which is not marked by label "E" (elementary) and where $i \in \{0, 1, 2, ...\}$. If this leaf does not exist then subgraph G' is the graph G of threats and the algorithm has come to its end.
3. The node U_i represents threat H_i.
   - 3.1. If the threat H_i cannot be divided into partial threats, then the threat H_i is an elementary threat, the node U_i is marked by label "E", and the next step is step 2.
   - 3.2. If the threat H_i can be divided, then this threat is divided into its partial threats H_j where $j \in \{1, 2, 3, ...\}$ according to requirement P. The following procedure is performed for each partial threat H_j.
     - 3.2.1. If the threat H_j has its node U_j in G' already, the subgraph is expanded by adding the oriented edge from U_i to U_j. The next step is step 2.
     - 3.2.2. If the threat H_j does not yet have its node U_j in G', a new node U_j is created, which is connected to the subgraph by oriented edge from U_i to U_j. The next step is step 2.

Note 1: The leaf is a node which is not an initial node for any edge.
Note 2: Indices i and j are ordinal numbers of threats or nodes.

In the first step, the initial subgraph of threats is defined, which will be expanded stepwise. The subgraph is formed by a single node, which represents the highest threat to security of the system.

In the second step, a leaf U_i is sought that has not yet been marked as an elementary node. This node represents threats H_i, which can be potentially divided into partial threats. In this way, an expansion of the subgraph could occur. If there is no such node, then all elementary threats have been found and the algorithm has come to its end.

In step 3, it is examined whether threat H_i can be divided into partial threats. If this is not possible (step 3.1) then the given threat is an elementary threat, which is expressed by label "E" of the respective node and we come to step 2. If the threat H_i can be further divided (step 3.2), then it is divided into partial threats H_j. Each partial threat is examined whether it has already been determined or whether it is a new threat.

In the case that the threat H_j has already been determined (step 3.2.1), then its node exists in the subgraph and therefore we only connect node U_i to node U_j. An example of the situation described above is the threat of computer theft, which is a partial threat in the threat of unavailability of hardware, in the threat of unavailability of software and also in the threat of unavailability of data. The next step is step 2.

In the case that the partial threat H_j is a new threat (step 3.2.2), then the subgraph of threats is expanded with the respective node U_j, which is connected to the edge from node U_i. The next step is step 2.

Unlike the methods in [1, 2, 3], the generated oriented graph G of threats need not be a tree. The fact that a superior threat is always represented by logic sum (OR) of partial threats and never by logic product (AND) of these threats is its other specific feature.

3. Threat analysis

Each node in graph G represents some threat. The leaves of G represent elementary threats H_j, which we also denote E_j to emphasize their elementariness. All the other nodes represent the so-called aggregated threats H_i, which due to the requirement P can formally be expressed as:

$$H_i = \bigcup_{j \in I} H_j, \tag{3}$$

where I is the set of numbers of all nodes that terminate the edges going out from U_i.

By iterating this formula, we can express every aggregated threat H_i by elementary threats E_j:

$$H_i = \bigcup_{j \in I} E_j, \tag{4}$$

where I is the set of leaf numbers of all paths that are going out from node U_i and terminate in a leaf.

We establish logic variable $h_i \in \{0, 1\}$ for each threat H_i, which we call threat realness. If $h_i = 1$, then the given threat is realistic and if $h_i = 0$, then the given threat is eliminated by some additional security measure, i.e. the threat is not realistic. In the same way, we define the realness of an elementary threat, which we mark e_i.

With the help of variables h_i and e_i, we can now analyze the influence of additional security measures. Due to the requirement P, we can express realness h_i of

aggregated threat according to (4) with the help of the Boolean logic from e_i:

$$h_i = \sum_{j \in I} e_j , \quad (5)$$

where the sum operation (i.e. +) represents logical sum (OR) and I is the set of leaf numbers j of all paths that are going out from node U_i and terminate in a leaf.

A generalization of the aggregated threat is the so-called combined threat C_k. This threat is some logical combination of elementary threats. Again, due to the requirement P we can express the realness of a combined threat c_k as a logical function F_k of elementary threat realness e_j:

$$c_k = F_k(\mathbf{e}), \quad (6)$$

where **e** is the realness vector of all elementary threats.

For example, analyze the threat that an attacker can obtain a firmware FW from a given device. This firmware is perpetually saved in memory chip U1 and after the device is switched on, the firmware is transmitted in encrypted form from U1 to block B1 inside the device. We denote this transmission P1. Key K from memory chip U2 is used for encrypting the transmitted firmware. Let us suppose that we have identified the respective elementary threats:

- threat E_1 that the attacker can read the content of U1 (i.e. obtain FW directly),
- threat E_2 that the attacker can monitor transmission P1 (i.e. obtain encrypted FW),
- threat E_3 that the attacker can read the content of U2 (i.e. obtain key K).

Thus, the threat H is real either in the case of realizing E_1 (the attacker finds FW from U1 directly) or in the case of realizing E_2 and E_3, when the attacker intercepts the transmission of encrypted FW (E_2) and also obtains the decrypting key (E_3). Then, we can formally express the realness h of threat H in the form $h = F(\mathbf{e}) = F(e_1, e_2, e_3) = e_1 + (e_2 \cdot e_3)$.

Threat tree methods [2, 3] can be used to construct combined threats C_k. In this case, however, the creator of the tree must only use elementary threats, which are generated by the algorithm described above.

In the end, we obtain m logical functions $F_1, F_2, ..., F_m$ for all m combined threats of the system under analysis. Arguments of these functions represent the realness of elementary threats. The system of functions $F_1, F_2, ..., F_m$ enables a very effective analysis of the realness of all combined threats and also enables judging the effectiveness of possible measures against elementary threats.

The following simple example illustrates possible usage of functions $F_1, F_2, ..., F_m$. Suppose two combined threats C_1 and C_2 which are represented by functions $F_1 = e_1 \cdot e_2 \cdot e_4$ and $F_2 = e_2 \cdot e_5 + e_3 \cdot e_4$. For this system of functions, we find all vectors $\mathbf{e}_s = (e_1, e_2, e_3, e_4, e_5)$ for which $F_1(\mathbf{e}_s) = F_2(\mathbf{e}_s) = 0$. The trivial solution $\mathbf{e}_0 = (0, 0, 0, 0, 0)$ represents the possibility to eliminate both combined threats by the elimination of all elementary threats. However, another possible solution $\mathbf{e}_1 = (1, 0, 0, 1, 1)$ represents possibility to eliminate both combined threats by the elimination E_2 and E_3 only. In this case, vector $\mathbf{e}_1$ represents more effective solution of countermeasures than vector $\mathbf{e}_0$. Then, general method can be following. We choose some suitable combination of countermeasures for each solution $\mathbf{e}_s$. In this manner, we obtain a certain combination of countermeasures for each solution. Finally, we choose the optimal (e.g. the cheapest) combination of countermeasures from all combinations.

4. Conclusions

The proposed method for threat analysis is suitable for complex and extensive systems. Its principle consists in finding the set of mutually independent elementary threats, from which combined threats are defined using the Boolean logic. By analyzing the obtained system of logical functions for combined threats, the significance of elementary threats can be established, which enables optimizing the choice of security countermeasures.

The proposed method is applicable to analyses of algorithms, protocols, devices and whole systems. It can be used for the threat analysis of systems that are being designated or modernized. The method also leads to the minimization of failures caused by the creator of threat graph, because the requirement for dividing threats into a complete set of mutually disjunctive threats minimizes the possibility of neglecting some threats.

References

[1] STAMATELATOS, M. et al.: Fault Tree Analysis with Aerospace Applications. NASA, Washington 2002.
[2] AMOROSO, E.: Fundamentals of Computer Security. Prentice Hall, Upper Saddle River 1994.
[3] SCHNEIER, B.: Attack Trees. Dr Dobb's Journal, Vol.24 (1999), No.12 (Dec.), pp. 21-29.
[4] MOORE, A. P. - R. J. ELLISON - LONGER, R. C.: Attack Modelling for Information Security and Survivability. Carnegie Mellon University, Pittsburgh 2001.

Karel Burda received the M.S. and PhD. degrees in Electrical Engineering from the Liptovsky Mikulas Military Academy in 1981 and 1988, respectively. During 1988-2004, he was a lecturer in two military academies. At present, he works at Brno University of Technology. His current research interests include the security of information systems and cryptology.