Graded Security Expert System

Abstract. A method for modeling graded security is presented and its application in the form of a hybrid expert system is described. The expert system enables a user to select security measures in a rational way based on the Pareto optimality computation using the dynamic programming for finding points of Pareto optimality curve. The expert system provides a rapid and fair security solution for a class of known information systems at a high comfort level.

Key words: Graded security, coarse-grained security analysis, Pareto optimal security evaluation

1 Introduction

Graded security measures have been in use for a long time in the high-risk areas like nuclear waste depositories, radiation control etc. [1]. Also in cyber security, it is reasonable to apply a methodology that enables one to select rational security measures based on graded security, and taking into account the available resources, instead of using only hard security constraints prescribed by standards.

It is well known that complete (100%) security of an information system is impossible to achieve even with high costs. A common practice is to prescribe the security requirements that have to be guaranteed with a sufficiently high degree of confidence for various classes of information systems. This is the approach of most security standards, e.g. [2]. However, a different approach is possible when protecting a critical information infrastructure against the cyber attacks – one may have a goal to provide the best possible defense with given amount of resources (at the same time considering the standard requirements). This approach requires a considerable amount of data that connects security measures with required resources and security measures with provided degree of security.

Practically, only a coarse-grained security can be analyzed in such a way at present, using a finite number of levels (security classes) as security metrics. This is a basis of the graded security methodology. This approach has been successfully applied in the banking security practice and included at least in one security standard [3]. The ideas of graded security are based on the US Department of Energy security model from 1999 [4] and its updated version from 2006 [5].

The graded security model itself is intended for helping to determine a *reasonable* set of needed security measures according to security requirements levels. However, in practice it can be the case that there are not enough resources to achieve the baseline. In this case it is still desirable to invest the limited amount

of resources as effectively as possible, i.e. to find and apply an *optimal* set of security measures.

The data required for estimating required resources and security measures can be presented in the form of expert knowledge in an extendable expert system. At present, this expert system can include at least the data that have been used in the banking security design, in particular in a branch of the Swedish bank SEB. Using an expert system has the advantage that it provides flexibility in selecting the required values for the security analysis – the values can be selected based on various input data, and even default values can be used in some non-critical places.

The present paper is organized as follows: the graded security model is presented in Section 2, the optimization method for finding a Pareto optimal curve depending on available resources is described in Section 3, and Section 4 gives a brief overview of the whole software system together with a demo example of security analysis.

2 Graded Security Model

In the present section we briefly explain the basic concepts of the graded security model: security goals, classes and measures as well as costs related to the security measures. We use integrated security metrics for representing the overall security of a system. We explain the way these entities are related.

Conventional goals of security are confidentiality, integrity, availability, and non-repudiation. In this presentation, that is based mainly on banking security, we use the following four slightly different *security goals: confidentiality (C), integrity (I), availability (A)* and *satisfying mission criticality (M)*. (The latter two are in essence two aspects of availability.) The model can be extended by including additional security goals. A finite number of levels are introduced for each goal. At present, we use four levels *0, 1, 2, 3* for representing required security, but the number of levels can vary for different measures. The lowest level 0 denotes absence of requirements.

Security class of a system is determined by *security requirements* that have to be satisfied. It is determined by assigning levels to goals, and is denoted by respective tuple of pairs, e.g. `C2I1A1M2` for the system that has second level of confidentiality C, first level of integrity I etc.

To achieve the security goals, some security measures have to be taken. There may be a large number of measures. It is reasonable to group them into *security measures groups*. Let us use the following nine groups in our simplified examples which are based on an educational information assurance video game CyberProtect [6]:

- user training,
- antivirus software,
- segmentation,
- redundancy,
- backup,

– firewall,
– access control,
– intrusion detection,
– encryption.

The number of possible combinations of security levels for all security goals is $4^4 = 256$. This is the number of different security classes in our case, see Fig. 1. A security class determines minimal required security levels for each group of security measures. *Abstract security profile* is an assignment of security levels (0, 1, 2 or 3) to each group of security measures. Hence, in the present example, we have totally $4^9 = 2621144$ abstract security profiles to be considered. The number of security measures groups may be larger in practice, e.g. 20. This gives a big number of abstract security profiles – 4^{20} for 20 groups. Knowing the costs required for implementing security measures of any possible level, one can calculate the costs of implementing a given abstract security profile.

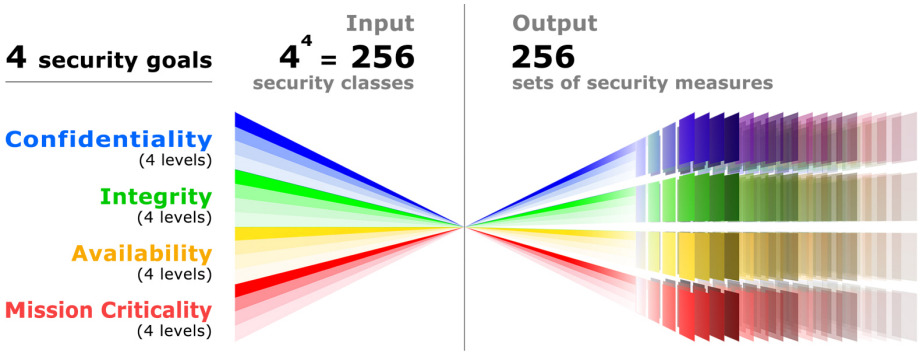

Fig. 1. Security classes of graded security model

After selecting security levels for a security measures group, one can find a set of concrete measures to be taken. For example, in the case of the security level 1 for the group "user training" the following measures have to be taken:

– New employees must be instructed for security – procedures and practice must be explained.
– An employee must know security related rights and obligations, must understand security practice, know about handling of passwords and keys.
– An employee must be instructed about security regulations and should be motivated to follow the regulations. Help about security must be available for all users of information systems.

This information is kept in the knowledge modules of the expert system of security measures, see Section 4.

It is assumed that, applying security measures, one achieves security goals with some confidence. The security confidence l_i is described by a numeric value

between 0 and 100 for each group of security measures $i = 1, \ldots, n$, where n is the number of groups.

We describe overall security of a system by means of an integrated security metrics – the security is evaluated by *weighted mean security confidence* S:

$$S = \sum_{i=1}^{n} a_i l_i \ ,$$

where l_i is security confidence of i-th security measures group, a_i is a weight of the i-th group, $i = 1, \ldots, n$, and

$$\sum_{i=1}^{n} a_i = 1 \ .$$

Information about costs, required security measures and confidence levels needed for calculations is presented in the expert system that will be described in Section 4.

3 Optimization Technique

Finding optimal amount of resources to be spent for security is considerably more complex problem than calculating resources required for implementing security measures of a given security class. First, a security class prescribes security requirements and respectively – spending of some minimally required amount of resources r_{min}. Applying expert knowledge, it is easy to calculate also resources r_{max} that can be optionally spent for achieving the maximal possible security level –

$$S_{max} = \sum_{i=1}^{n} a_i l_{max\, i} \ ,$$

where $l_{max\, i}$ is maximal security confidence of the i-th group of security measures.

Applying some resources between the values r_{min} and r_{max}, one can get better security in a rational way. We have an optimization problem with two goals: to minimize resources on the interval $[r_{min}, r_{max}]$ and to maximize security, preferably guaranteeing the levels prescribed by a given security class. We are going to solve this problem by finding the abstract security profile that has maximal value of a fitness function given by the weighted mean security for a given value of resources. Repeating this calculation for sufficiently many values of resources on the interval $[r_{min}, r_{max}]$, we get a Pareto optimal solution of the problem expressed by a Pareto optimality tradeoff curve of the form shown in Fig. 2. Finally, the calculated optimal abstract security profile is compared to the concrete security profile prescribed by the security class – security levels should not be less than prescribed.

The exhaustive search of optimal solutions for q possible values of resources, n security measures groups and k security levels requires testing (calculating

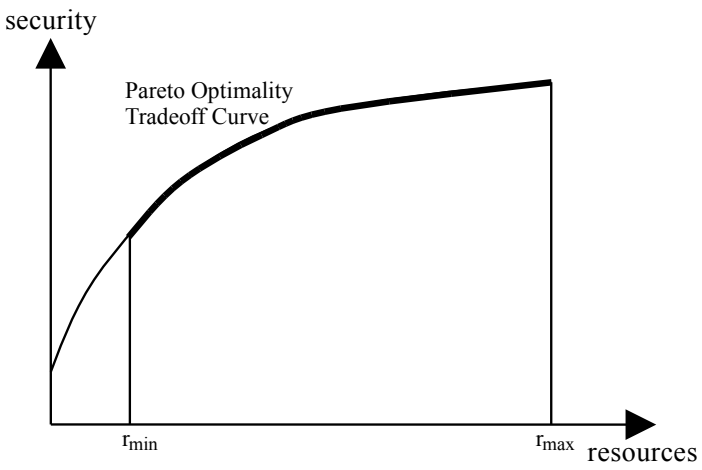

Fig. 2. Search of optimal security along resource dimension

weighted mean confidentiality) of qk^n points. Building optimal solutions gradually, for $1, 2, \ldots, n$ security measures groups enables us to use discrete dynamic programming, and to reduce considerably the search time. Indeed, the fitness function S defined on intervals from j to k as

$$S(j,k) = \sum_{i=j}^{k} a_i l_i$$

is additive on the intervals, because from the definition of the function S we have

$$S(1,n) = S(1,k) + S(k,n) \; .$$

This means that one can build an optimal resource assignment to security measures groups gradually, as a path in the space with coordinates x_1, x_2, where x_1 equals to the number of security measures groups that have got resource (i.e. $x_1 = k$) and x_2 equals to the amount of used units of resources ($1, 2, \ldots, 1000$ in our example). Figure 3 shows a search step, where known optimal partial solutions (assignments of resources to already tested security measures groups) are the paths from initial state (where no resources are assigned) to intermediate states $s_1, \ldots, s_n$. The aim is to find one step longer optimal paths from a to the states $t_1, \ldots, t_m$ that follow the states $s_1, \ldots, s_n$. This can be done for each security measures group $i = 1, \ldots, n$ by trying out all possible continuations of the given partial optimal paths to $s_1, \ldots, s_n$ as shown in Fig. 3. This algorithm requires testing of q^2kn points (q is number of possible values of resources, k is the number of security levels, n is number of security measures groups).

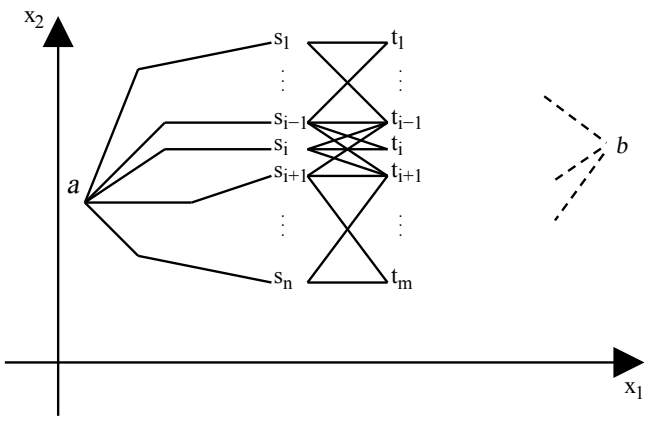

Fig. 3. Resource assignment by means of discrete dynamic programming

4 Security Expert System

A hybrid expert system with visual specification language for security system description has been built on the basis of a visual programming environment CoCoViLa [7]. The system includes knowledge modules (rule sets) in the form of decision tables for handling expert knowledge of costs and gains, as well as for selecting security measures for each security group depending on the required security level. Other components are an optimization program for calculation Pareto optimality curve parameterized by available resources, and a visual user interface for graphical specification of the secured system, visual control of the solution process through a GUI, and visualization of the results. These components are connected through a visual composer that builds a Java program for each optimization problem, compiles and runs it on the request of the user, see Fig. 4.

Let us explain the usage of the expert system on the following simplified example. We have nine security measures groups as given in Section 2. Two groups – "user training" and "encryption" – have specific values of cost and confidence related to security levels that must be given as an input. We can use standard values of cost and confidence given in the expert knowledge modules for other groups. We have to solve the problem in the context of banking and can use resources measured in some units on the interval from 1 to 70. The security class `C2I1A1M2` is given as an input. The expected outcome is a graph that shows the weighted mean security confidence depending on the resources that are used in the best possible way. The graph should also indicate whether the security goals specified by the security class can be achieved with the given amount of resources. Besides that, the curves showing security confidence provided by user training and redundancy must be shown.

The visual composer is provided by the CoCoViLa system that supports visual model-based software composition. The main window of the expert system

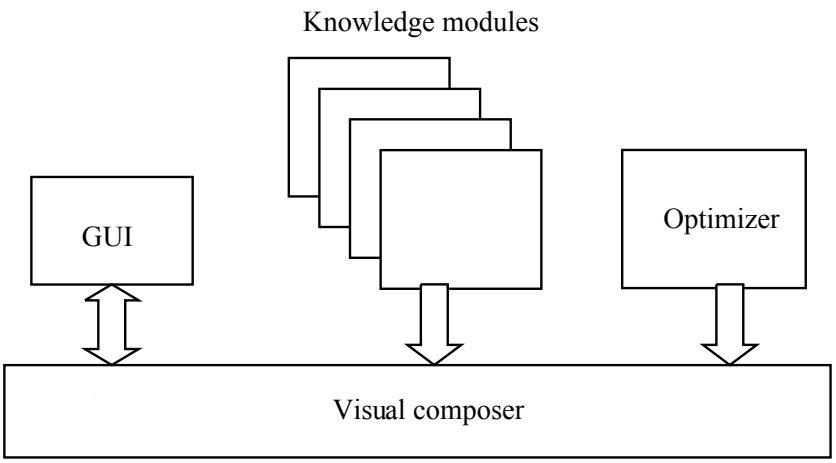

Fig. 4. Graded security expert system

shown in Fig. 5 presents a complete description of the given problem. It includes also visual images of components of the expert system and a toolbar for adding new components, if needed. In particular, new security measures groups can be added by using the third and fourth button of the toolbar. Besides the security measures groups there are three components – Optimizer, SecClass and GraphVisualizer – shown in the window. The components in the main window can be explicitly connected through ports. This allows us to show which values of security should be visualized ("user training" and "redundancy" in the present case) etc. There are two different views of security measures groups – "user training" and "encryption" that have explicit values of costs and confidence given as an input. Other groups use the standard values of costs and confidence given in the expert knowledge modules as specified in the problem description. The SecClass component is used for specifying security goals. During computations the component also evaluates the abstract security profiles calculated by the Optimizer against the actual security requirements using a knowledge module from the expert system.

In Fig. 6 there is a window showing the optimization results. The first curve (Confidence) represents the optimal value of weighted mean security confidence depending on the resources that are used in the best possible way. This curve is further divided into four parts to visualize to which degree the optimal result satisfies the security requirements given by the security class. The first part (thin black line) indicates the interval of resources where none of the four (in our example) security goals can be achieved. The second part (thin grey line, three separate segments) shows that at least one of the security goals is satisfied while also at least one is not. The third part (thick black line) represents the amount of resources that, when used optimally, would result in satisfying the requirements exactly. One should note that this coincidence of the optimal security profile

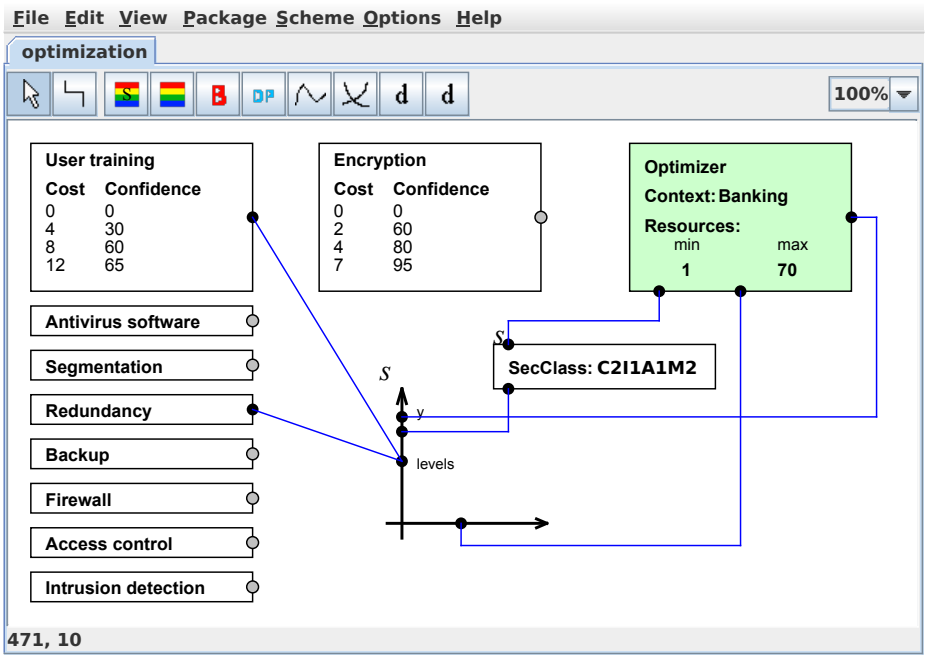

Fig. 5. Problem specification window

and the security requirements does not always exist. The last part of the graph (thin black line, again) shows the amounts of resources that are more than is strictly needed to satisfy the requirements. It is interesting to notice that on the interval of costs from 36 to 45 units it is possible to satisfy all security goals, because already spending 34 units enables one to do this. However, the solutions with highest values of the weighted mean security confidence do not satisfy all security goals on this interval.

The lower graphs indicate (on the right scale) the optimal levels of two measures groups corresponding to the given amount of resources. These graphs are not necessarily monotonic as can be seen in this example at the resource values 35 and 36. When there are 35 units of resources available it is reasonable to apply the measure "user training" at level 2. Having one more unit of resources better overall security confidence level is achieved by taking all resources away from "user training" and investing into the "redundancy" measures group to achieve level 3.

5 Conclusions

The advantage of the expert system of the graded security is that it provides a rapid security solution at a sufficiently high although not 100% confidence level. Based on our previous experience, the graded security expert system allows a

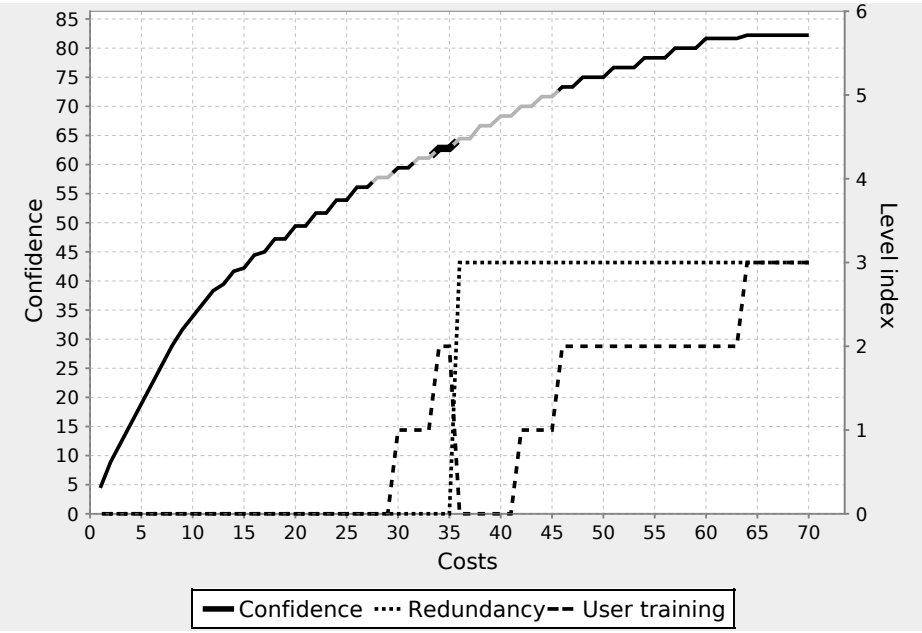

Fig. 6. Solutions window

typical security solution to be developed within approximately 8 hours, with about half the time spent on security class identification and the other half on analyzing available resources, accepted security risks, attack costs and other optimization variables. Our method reduces the time for analysis and provides a Pareto optimal solution. It includes:

- graded security selection procedure that yield the security measures for a given security class;
- high-level analysis of usage of resources for information security and accepted risks based on advanced optimization technique.

We understand that wider application of this method will depend on the availability of expert knowledge that binds costs and security confidence values with taken security measures. This knowledge can be collected only gradually, and will depend on the type of the critical infrastructure that must be protected.

Acknowledgements

...

References

1. Kang, Y., Jeong, C. H., Kim, D. I. Regulatory approach on digital security of instrumentation, control and information systems in nuclear power plants. Korea In-

stitute of Nuclear Safety. Daejeon, Korea. http://entrac.iaea.org/I-and-/TM_IDAHO_2006/CD/
2. German Federal Office for Information Security (BSI). IT Baseline Protection Manual. 2005. http://www.bsi.de/gshb/
3. Estonian Information Systems Three-Level Security Baseline System – ISKE ver. 1.0.
4. U. S. Department of Energy, Office of Security Affairs. Classified Information Systems Security Manual. 1999.
5. U. S. Department of Defense. National Industrial Security Program Operating Manual (NISPOM). 2006.
6. U. S. Department of Defense, Defense Information Systems Agency. CyberProtect, version 1.1. July 1999. http://iase.disa.mil/eta/product_description.pdf
7. Grigorenko, P., Saabas, A., Tyugu, E. Visual tool for generative programming. - ACM SIGSOFT Software Engineering Notes, 2005, 30, 5, 249-252.