Quantitative assessment of enterprise security system

Ruth Breu, Frank Innerhofer–Oberperfler
Research Group Quality Engineering
Institute of Computer Science, University of Innsbruck
Technikerstr. 21a, A-6020 Innsbruck, Austria
Email: {ruth.breu, frank.innerhofer-oberperfler}@uibk.ac.at

Artsiom Yautsiukhin
Dip. Informatica e TLC
Universit degli Studi di Trento
via Sommarive 14, I-38100 Povo (TN), Italy
Email: evtiukhi@dit.unitn.it

***Abstract*—In this paper we extend a model-based approach to security management with concepts and methods that provide a possibility for quantitative assessments. For this purpose we introduce security metrics and explain how they are aggregated using the underlying model as a frame. We measure numbers of attack of certain threats and estimate their likelihood of propagation along the dependencies in the underlying model. Using this approach we can identify which threats have the strongest impact on business security objectives and how various security controls might differ with regard to their effect in reducing these threats.**

I. Introduction

The increasing dependency of core business processes and activities on information technology has transformed IT security management from an issue of the IT department to a boardroom issue [27]. Due to the different focus on security by business managers and more technically oriented security professionals [23] we consider it important to employ methods and models that bridge those differing views. With this problem in mind an approach that models business and technical oriented concepts of an enterprise system was developed [6]. The approach is based on the ideas of enterprise modeling and enterprise architectures [3].

The approach that far consisted of metamodels that define the concepts and interrelations to specify an enterprise model and to depict security related concepts and information. In addition a guided process that attributes responsibilities to various roles and their collaboration has been defined [12]. However until now, the approach was solely qualitative in nature and a quantitative analysis of the models was not yet available. In this paper we present the extended approach introducing concepts and methods that allow quantitative assessment of security.

A quantitative extension of the approach is useful for a variety of reasons. First of all, a quantitative measurement of security provides a more fine-grained way of assessing the risks to security. This allows identifying the weakest link in an enterprise security system and those risks that have the highest impact on the business. Second, it provides a basis to measure the value of intended security investments calculating their effect from a business point of view.

For this purpose we introduce a quantitative assessment that is based on the following measures. First of all, we introduce a measure that quantifies the ratio of attacks that results in successful breaches of security requirements. For this purpose we count the number of attacks for a specific period of time. Second, we quantify the propagation effect of successful attacks. Finally, when the numbers of violations of security requirements directly derived from a business security objective are found we compute the losses for the enterprise caused by failing to fulfil the objective.

The article is organized as follows: In section II we describe the Enterprise Model that is the basis of our approach. In section III we describe the Security Model that contains the security relevant information. In section IV we describe our approach of quantitative assessment of security. In section V we give an overview of our ideas how to use the quantitative measures to guide security investment decisions. Finally in the last sections we discuss strenghts and weaknesses of our appoach, give an overview of related work and outline future work.

II. Enterprise Model

The proposed assessment method is based on the modelling of enterprise IT architecture which has been presented in [6], [12]. In this paper we just outline the main concepts of the approach which are relevant for the current paper. Those who are interested in a more detailed discussion about the modelling we refer to [6], [12].

The enterprise meta-model contains three layers (Figure 1):

- **Business Layer.** Business layer contains business oriented artifacts. First of all it contains *Organizational Unit* which can be the enterprise as a whole, business units or departments. *Roles* are collective sets of responsibilities and obligations imposed on an employee which plays this role in the enterprise. Role, which is a part of an Organizational Unit, is involved in one or several business processes of the enterprise. *Business process* is a predefined sequence of activities leading to accomplishing of a goal. Business process may be seen as a hierarchy of sub-processes or atomic activities. *Information Objects* are the information which is used by business processes. In this paper we do not deal with Organizational Units

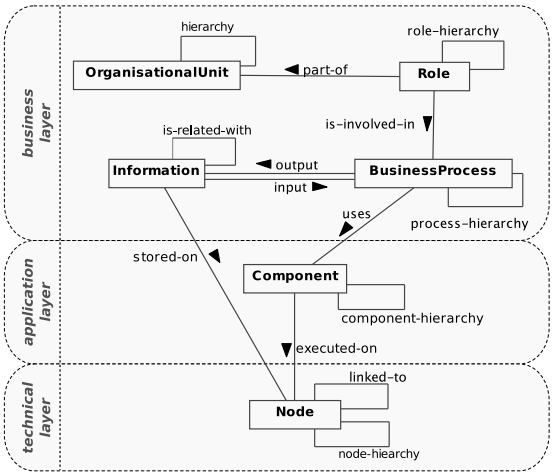

Fig. 1. Enterprise meta-model

and Roles assuming that these artifacts cannot be impacted by security breaches.

- **Application Layer.** On the application layer we only have Components. *Components* are the applications used by some business process (by one of its activities).
- **Technical Layer.** Technical layer contains nodes. *Node* is a software and hardware set which provide all required services for Components to operate or Information Objects to be stored. For example, node can be a server with installed Windows Server 2003 and Oracle database. Nodes can be hierarchically modelled. For example, a node may be seen as a collection of simple nodes in order to simplify an analysis.

In general an enterprise model may be presented as it is shown in Figure 2.

Definition 1: Model elements in scope of this paper are Business Processes, Information Objects, Components and Nodes. *Model dependencies* indicate the relations between model elements. *Domain* indicates the areas of responsibility of a specific domain owner. Note, that we distinguish notions of domain owner and role here. *Domain owner* is an entity managing a set of model elements (usually at the same level). *Dependency Graph* is a graph formed by model elements as nodes and dependency relations as edges. The graph may start from every model element (called root element) and spread according to dependency relations. The Dependency Graph defines the scope of security analysis and the dependency relations are used as a frame to propagate security relevant information throughout the Enterprise Model.

Example 1: Consider a banking company as an example of such modelling (Figure 3). In the figure we joined several dependency relations for the sake of simplicity. The company has three business processes which correspond to three main services provided by the bank: loan issuance, money transfer, money investment. At the business layer we also have three information resources which are stored in the system and used by the processes. Note, that the application layer contains

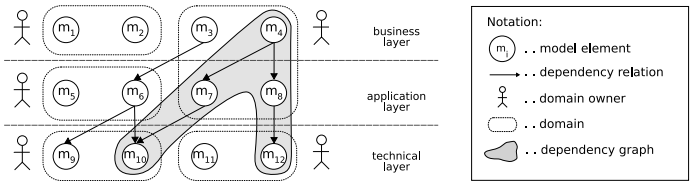

Fig. 2. General enterprise model

only the applications which are used by the processes and the software which is needed only to execute these applications is considered as a part of a node. For example, authentication engine (component) requires a PC with installed Windows Server 2003 (node).

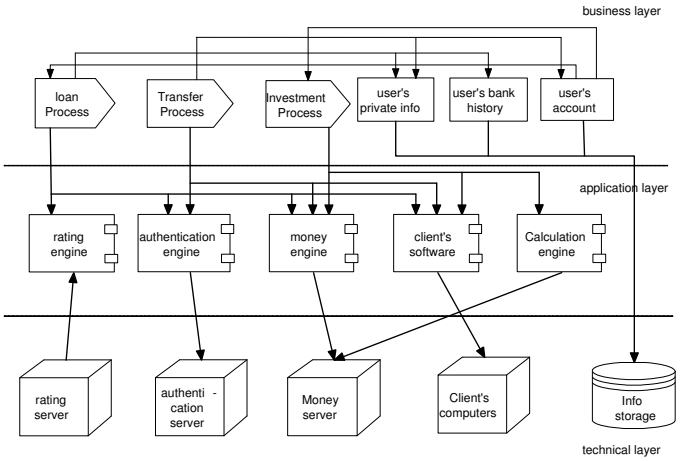

Fig. 3. Enterprise Model of a bank's IT architecture

III. Security Model of an Enterprise

We need the enterprise model for identification of interdependencies between model elements. This information is used for elicitation and specification of security requirements for these elements. The elicitation starts with identification of a business security objective (BSO) (or business security goal). Domain owners ascertain if the BSO is relevant for some of the elements in their domains (typically it is done for business model elements). These model elements are root elements of Dependency Graphs which contains all elements contributing to satisfaction of the business security objective.

The business security objective is then specialized for the model elements to specify security requirements [18]. It is advisable to have several requirements if the impact on the organization caused by not fulfilling the requirements varies too much. This facilitates the assessment of the monetary impact on the BSO. Then the security requirements are decomposed further according to the Dependency Graphs to specify the security requirements for dependant model elements. Requirements may be decomposed even on the same level to make the analysis more fine-grained if it is necessary. The received graph is called Security Requirements Graph and shows how

security requirements for high level model elements impose requirements for lower elements.

Example 2: In Figure 4 we show a Security Requirements Graph for our banking running example. Business security objective is “Prevent frauds” which is relevant for all business processes and “user’s account” object. For the sake of simplicity, we consider only the frauds which occur during loan origination process. Such view may have a manager which is responsible for the process (domain owner). Requirements for other business model elements are decomposed by corresponding domain owners.

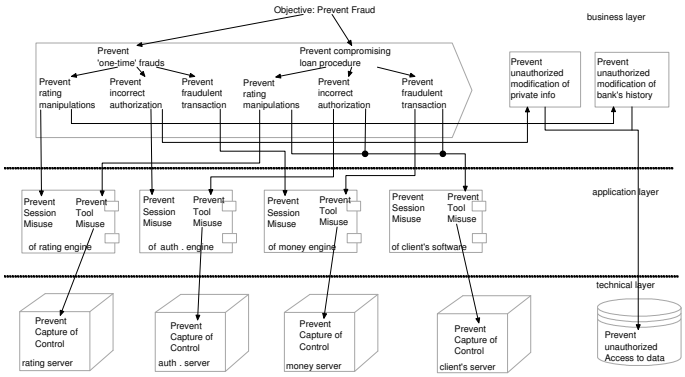

Fig. 4. Requirements decomposition from loan process perspective

We elicited two requirements for loan process from the BSO. The idea behind this is that losses caused by a one-time fraud (or just several frauds), e.g., caused by stolen authentication information, are lower than the losses caused by violating the whole procedure, e.g., maliciously modified money transaction program operating for several months. These requirements are decomposed further on business layer for more fine-grained analysis.

Then the decomposition goes to application layer where either a session may be forged or the tool may be modified to operate maliciously. The latest event may occur if an attacker has sufficient access rights at the corresponding server. At last the corresponding requirements are assigned to nodes.

One more important point which can be seen in Figure 4 is that there are inner dependencies on business layer. The dependencies appear because the loan process uses some information resources as it is shown in Enterprise Model. This means that malicious change in an information resource (e.g., client’s history) may lead to incorrectly issued loan.

At this point one can see that the graph may have circles. On the business level the circle can be formed by ”input” and ”output” dependencies between processes and Information Objects. The simplest circle means that a business process compromises an Information Object and then uses it and, as a consequence, compromises itself. Although this situation is possible (if one instance of the business process compromises the object in order to violate another instance of the same business process) we consider such situation as highly improbable. Therefore, we assume that such circles should not be taken into account. In the graph this means that the dependency, which leads *from* one requirement to another one that in its turn influences the former requirement through some dependency chain, is not included in the Security Requirements Graph.

On the technical level the circle can be formed by ”link-to” bidirectional relations. The simplest example is network connections: a workstation is connected to a server and vice versa. Analysis taking into account such relations can show how a threat (e.g., worm) propagates through technical layer. On the other hand, data for our analysis (Section IV) is more likely to be taken from security logs of the nodes and thus we will already have the total number of attacks for the node. This means that we do not have to consider the dependencies between nodes because the threat propagation is already taken into account by premise data.

IV. Quantitative assessment of the model

The next step in the security analysis process is the identification of potential threats and their risk assessment. For this purpose the Security Meta Model (see Figure 5) defines risk as a triple of a model element of the Enterprise Model, a security requirement attached to the element and a threat that targets the specific model element and breaches the related security requirement. We therefore have a specific notion of Risk that is defined as follows:

Definition 2: A *Risk* is defined as any threat that targets a specific model element and may result in the violation of a security requirement.

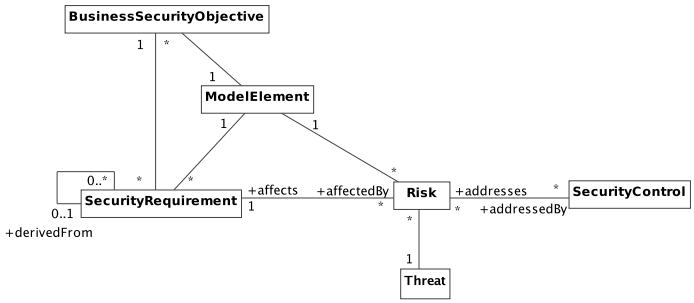

Fig. 5. Security meta-model

Since the main security goal is to prevent bad events happen we can reformulate our goal in the negative way [19], [26]: *analyze how much the enterprise looses because the BSO fails.*

First of all we should find the number of violations of a BSO. For this purpose the requirements should be inverted. The inverted requirements become negative events (threats) which may occur in the system and violate the BSO. We also should invert dependency relations to capture the fact that now low level threats may cause satisfaction of high level threats (or failure of the higher level requirements). We call this reversed Security Requirements Graph as Threat Graph.

Example 3: A Threat Graph for our running example is shown in Figure 6. Note that the inversion has been done in such a way as to calculate numbers of threats occurrences, our metric for aggregation. For example, the requirement “prevent

'one-time' fraud" is transformed to the threat "'one-time' fraud".

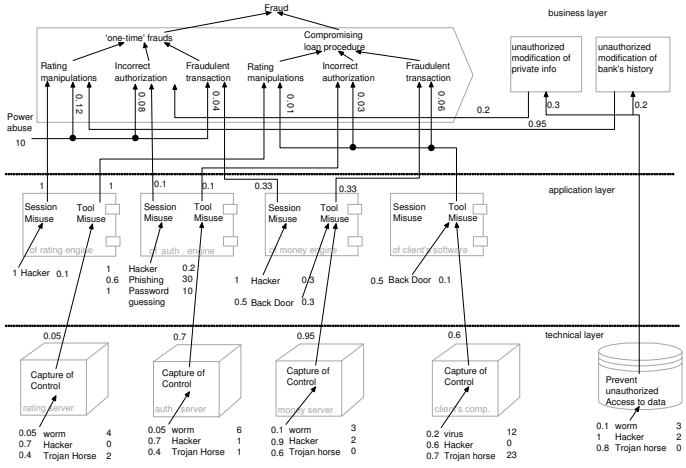

Fig. 6. Threat Graph with threats and weights

To finalize construction of the Threat Graph concrete threats should be identified.

Definition 3: Concrete threats are context-independent negative events which lead to occurrence of a specific (context-dependent) threat in Threat Graph.

Examples of such concrete threats are: viruses, hacker attacks (exploiting buffer overflow, SQL injection, etc.), theft, power abuse and so forth. Threats can be attached to the requirements at *each* of three layers. Some threats may occur on the business layer (e.i., procedural), others may target to specific application, the rest affect the business security objective by compromising nodes.

To support the definition and identification of potential threats — apart from the negation of our security objectives and security requirements — we employ standards and catalogues of typical IT security threats [13], [7], [9].

Example 4: An example of procedural threat is power abuse[1] of the following kind: a dishonest employee authenticates a bogus customer. Exploiting an SQL injection bug by a hacker in the money transfer software is an application layer threat. On the technical layer a trojan horse may affect the operating system on the money server and send root login/password pair to an attacker giving him access to the programs running on it, its memory and stored data.

The next step in the preparation of the Threat Graph for the analysis is assigning weights to the edges. A weight attached to an edge denotes the probability for a target threat to occur if a source threat realizes.

This weight may be seen as a multiplication of two components: the attractiveness of a target goal for an attacker and impact of the source threat to the target threat. In other words, the former component is the probability that the attacker will choose to compromise *this* target requirement (will realize *this* target threat). The later component is the probability that the attacker and will be able to realize the target threat after achieving some results (source threat) independently from other possible attack paths.

Example 5: It has been defined that only 60 % of capturing of control over a clients's node lead to misuse of client's software. So, a bit more than a half of successful attacks in technical layer will propagate further.

Example 6: Note, that the contribution of misused client's software to rating manipulation is very small. This can be explained by the fact that for an attacker gaining some benefit from compromising this activity is much more troublesome rather than compromising money transaction. Therefore, it is less probable that the attacker which was able to compromise client's software attacks the rating activity.

Formally the graph can be seen as follows:

Definition 4: A *Threat Graph* TG is a triple $\langle T, E, L\rangle$ where T is a set of threats and E is a set of dependency edges. Each dependency edge is an ordered pair $\langle t, t'\rangle$ from an arbitrary threat $t \in T$ (*source threat*) to a threat $t' \in T$ (*target threat*). L is a labelling function which assigns weights to the dependency edges. If we have a domain $\mathcal{D}$ then the labelling function is: $L : E \mapsto \mathcal{D}$.

Finally, we should assign values to leaf nodes of the Threat Graph. The values show how many successful attacks (realized threats) are expected.

Example 7: In Figure 6 we show the Threat Graph ready for the analysis: with concrete threats, assigned weights and identified expected numbers of attacks in 10 years. We have chosen 10 year period in order not to operate with very small numbers assuming that the bank has a very good protection. Note, that this is done for convenience only and the same analysis can be done for the banking scenario counting risks for 1 year period.

We did not draw separate edges from concrete threats to reduce the complexity of the picture. Therefore, the expected numbers of threats are depicted on the right hand of the threats and their contributions to target threats (i.e., weights) are shown on the left.

In this example we assume that all inner hierarchical relations have weights equal to 1. See the relations at business layer. This means that once an activity is compromised - the whole business process is compromised. In fact it may be the situation when this assumption does not hold. For example, the contribution will be less than 1 if the failure can be noticed lately while the process is being fulfilled by successful applying the Separation of Duty principle.

Now we can calculate the number of violations for *all* nodes in the Threat Graph. This can be done by multiplication of values of source threats by corresponding weights. If there are several source requirements contributing to a target threat (several incoming edges) the received numbers are summed up. In other words, for each target threat we apply a weighted function to compute the total expected number of occurrences.

[1]Although "power abuse" also may have a long term impact on the BSO we do not consider such case here for the sake of simplicity. On the other hand, if an analyst wants to take this fact into account he should simply assign this concrete threat to the long term impact as well.

Algorithm 1 Calculation algorithm (informal)

Require: $TG = \langle T, E, L \rangle$: Threat Graph;
N_{conc} : number of concrete threats;
Ensure: $N[]$: real; {Number of threats}
1: Assign premise numbers of attacks (N_{conc}) to leaf threat nodes
2: Add concrete threat nodes to the working set
3: **while** working set is not empty **do**
4: take one node from the working set
5: **for** each outgoing edge from the node **do**
6: **if** all source nodes for the new node are visited[2] **then**
7: compute weighted function for the new node
8: add the new node to the working set;
9: **end if**
10: **end for**
11: **end while**

Algorithm 2 Calculation algorithm (formal)

Require: $TG = \langle T, E, L \rangle$: Threat Graph;
N_{conc} : number of concrete threats;
Ensure: $N[]$: real; {Number of threats}
1: $N[T_{conc}] := N_{Leaf}$;
2: HEAP-insert(T_{conc})
3: $SOURCE[T] := |Incoming(T)|$;
4: **while** HEAP-nonempty **do**
5: HEAP-extract(t'); {randomly}
6: **for** $\langle t', t \rangle \in Outgoing(t')$ **do**
7: decrement(SOURCE[t]);
8: **if** $SOURCE[t] = 0$ **then**
9: $N[t] = 0$;
10: **for** all $t'_i.\langle t'_i, t \rangle \in Incoming(t)$ **do**
11: $N[t] = N[t] + L(\langle t'_i, t \rangle) * N[t'_i]$;
12: **end for**
13: HEAP-add(t);
14: **end if**
15: **end for**
16: **end while**

This procedure starts from the lowest threats and continues while the numbers of threats are found.

Example 8: Lets compute the number of misuses of client's software. According to the calculations on the technical layer it has been found that by various means an attacker may receive control over a computer of an employee 18.5 times in 10 years. After the attacker got the control over the node he can misuse the banking software by modifying it or by installing a malicious program which will change data in the memory while the application is working. Of course, not all attackers can do that. We assume that only 60% of attackers are able to modify the client's banking software. Using this information and the knowledge about possible back door attack we can calculate the number of misuses of the client's banking software: $0.1 * 0.5 + 18.5 * 0.6 = 11.15$. This number can be used for further aggregation.

According to the assumptions that the Threat Graph does not contain circles (see the discussion in Section III) we create the Algorithm 1 for calculation of the number of violations of requirements for root elements. Algorithm 2 is a formal version of Algorithm 1.

When the top threats, the ones corresponding to the requirements for root model elements, are reached and numbers (rates) of occurrence (ARO) for them are calculated monetary risk assessment can be conducted. The received numbers are multiplied by the corresponding average single losses expectancy (SLE) and the results are summed up. The final value is the expected loss for organization caused by failing to fulfil the specific BSO. In other words, we use the formula for classical Risk Assessment calculating Loss Expectancy (ALE)[3] [11]:

$$ALE = ARO * SLE$$

We do this multiplication at the end of the aggregation because the monetary impact on business security objective can be determined only when we know the numbers of requirement violations which are directly connected with the objective. Otherwise, we will not be able to separate the impact on various objectives.

Example 9: Knowing that number of 'one-time' loan frauds is 8 and that each fraud costs the bank approximately 5 000$ we can calculate that because of this threat the bank looses 40 000$ in 10 years. On the other hand the number of loan procedure compromised is 2 but the loss of such threat occurrence is in average 80 000$. This means that because of not fulfilling this requirement properly the bank looses 160 000$ per 10 years. Totally, loan frauds cost the bank 200 000$.

V. Security Controls

Using the Threat Graph one can also calculate the benefits from implementing countermeasures. Countermeasures in our model may influence the graph in the following two ways: (i) reduce number of concrete threats (ii) decrease the weights of a dependency leading to the threat. In the first case we should reduce the number of occurrences of the concrete threats the countermeasure mitigates. In the second case the countermeasure reduces the probability of a source threat to propagate further successfully.

Example 10: Pre-employment screening strategy [17] (evaluating past behavior, habits, social status of a candidate before the employment) reduces the number of untrustworthy employees in the organization and, therefore, reduces the number of power abuses by 30%. On the other hand, using iTAN[4] instead of TAN reduces the probability to compromise a transaction by attackers who obtained authorization information by phishing by 60%.

[2]Here "visited" means that the node has been added to and extracted from the working set

[3]Letter 'A' in the classical formula stands for 'annualized' (ALE and ARO). In fact the period of the analysis is not important and 'A' may be left out but we did not remove it just to leave the classical view of the formula.

[4]TAN is a Transaction Authorization Number. Client has a set of such numbers any of which can be used to confirm a transaction. iTAN is an indexed Transaction Authorization Number. The set of such numbers is similar to the set of TANs but now to confirm a transaction the client has to enter a specific number requested by the bank.

According to this change(s) the expected losses should be recalculated. This calculation does not require recomputing the whole Threat Graph but just the part which has been changed. Such analysis can be done for several possible countermeasures. The received reductions can be used for choosing the most appropriate selection.

Countermeasures also have a cost of deployment and maintenance. Therefore, it is advisable to do a cost-benefit analysis [11] to select the best solution. Two points should be taken into account while conducting the cost-benefit analysis using Threat Graph. By now we considered the impact on one BSO only. On the other hand, the losses could be on lower levels as well. Although failed authentication procedure costs to the enterprise much more than damaged authentication software (which has to be patched or re-installed) the latter also causes some losses. In order have the complete picture of losses *all* these impacts should be summed up. This means that at each reduced threat in the Threat Graph the reductions should be calculated and summed up. Note, that these are losses for a specific threat to a specific model element only. This should facilitate the estimation of the monetary impact.

Example 11: A security training program for bank's employees will reduce the number of viruses on client's computers by 40% (less employees will open attachments in untrustworthy mails). These means that there will be 11.1 successful virus attacks instead of 18.5. As a consequence there will be 7.4 successful capturing of client's computer less, 4.44 tool misuses and 0.444 frauds less. If the costs of the impacts are 500 $, 1000 $ and 100 000 $ then the countermeasure will reduce losses of the bank (for loan process) by $7.5*500+4.44*1000+0.444*100000 = 52\ 590$$.

Another point here is that a countermeasure could reduce occurrences of the threats which impact another BSO. To have a complete picture for cost-benefit analysis the Threat Graph constructed for other business security objectives should be evaluated.

Example 12: Introducing a new training program strategy costs the bank 120 000 $ per 10 years. On the other hand it reduces losses of the bank for loan process by 52 590 $ protecting it against fraud and by 41 000 $ protecting it against loss of availability and 57 000 $ protecting it against revealing of confidential information. In other words, the benefit from the countermeasure is 52 590 + 41 000 + 57 000 - 120 000 = 30 590 $.

VI. Related works

Recently a lot of attention has been devoted to assessment of security. One of the key problems here is the identification of a suitable indicator. Well-known SSE-CMM [28] uses maturity level (a value from 1 to 5) as an indicator of maturity of the security process. Some approaches recommend checking compliance with a security standard (e.g. ISO 17799 [13]) often using percentage of compliance as a metric for the evaluation [15].

Taking a more technical view (e.g., vulnerabilities) some authors try to calculate a "mean-time-to-breach" indicator [20], [22]. The most popular approach nowadays is applying risk analysis for security evaluation [8], [29] which is based on economical assessment. In our approach we propose a way to perform the analysis in a more fine-grained way then classical Risk Assessment allows which more thoroughly indicates the losses of an enterprise caused by a concrete threat for a specific BSO.

From the discipline of security management a variety of approaches underline the importance of managing information security from a business point of view. Some of these approaches are model-based and/or quantitative and will be related to our work.

An approach that approaches information security from a business perspective is the OCTAVE approach [2]. The difference to our approach is that we employ modelling techniques to visualise the relevant assets and their dependencies using an enterprise model. In the authors opinion the OCTAVE approach is very useful to identify the relevant assets of an organisation from a business perspective and we therefore see it complementary to our approach. The assets and technical artefacts identified with the OCTAVE approach could be modelled and analysed using our approach.

An approach that is following a model-based risk analysis is CORAS [1], [5]. While the CORAS approach uses models mainly for descriptive purposes we employ models to systematically understand and analyse the dependencies and interrelationships of business and technical objects of an enterprise.

Jürjens [16] has developed the UMLSec approach for model-based security engineering based on UML. UMLSec is an extension of UML that can be used to express security relevant information in UML diagrams of a system. The approach is mainly targeted towards secure system development while we use a enterprise modeling to analyse functional dependencies between business and technical artifacts.

Suh and Han [30] use a business model to identify business functions in order to evaluate the relative importance of information assets for these functions. Suh and Han focus solely on the security requirement of operational continuity. For this purpose they use a measure to denote the relative importance of technical assets for business functions.

Zambon et al. [33], [32] are also following an enterprise modeling approach to identify functional dependencies between business processes and technical systems. They are also aggregating the quantitative measures along the functional dependencies of the elements of an enterprise model in a very similar way like we do. They differ with their focus on downtime and temporal aspects of availability, while we are using the number of attacks as the key quantitative concept.

Our work is based on analysis of an Threat Graph which may be considered as a special type of an attack tree. Similar to our idea the construction of attack graph starts with a goal of an attacker (can be seen as a reverse business security requirement). Then the goal is decomposed on smaller objectives which the attacker should fulfill to achieve his goal. The decomposition continues until simple steps are reached [24],

[21], [19]. In contrast to this subjective way we propose a way of constructing Threat Graph using an enterprise model. We also provide a qualitative analysis which is based on a well-known risk assessment technique. In contrast to [4] we first aggregate values (number of attacks in a period of time, which is similar to Annual Rate of Occurrences) and then estimate the impact for business security objectives rather than consider threats separately without linking them with the objectives.

Clark et. al. [10] also aggregated risk levels using an attack tree in order to evaluate the impact on enterprise's mission (business goal), but for aggregation they used number of existing vulnerabilities on leaf nodes treating them as risk measures which differs from classical Risk Assessment.

Although we work on a level of requirements and threats and do not consider low level vulnerabilities our approach have something in common with attack graphs [22], [31], [14], [25]. In the work we implicitly use the causes and consequences of threats occurrences to build the Threat Graph while contracting Security Requirements Graph. On the other hand, our approach is top-down one (we start with a BSO and then refine the objective) when attack graph technique is a bottom-up approach (starting with a premise conditions it shows how an attacker can reach his goal).

VII. Discussion

In this section we will give a short discussion about the general weaknesses and strengths of this approach.

The main advantage of the proposed assessment method is that it provides a way to quantify the impact of a security state of a specific element in the enterprise on a business security objective. The analyst may choose not only the technology which can mitigate the impact but also a concrete place where the countermeasure should be installed and thus more wisely distribute limited security resources. For example, if a more rigid software patching strategy is required and human resources are all already in use then it will be better to relocate some staff/time from rating server, which impacts 'prevent fraud' objective not very significantly, to money server, which has the greater impact on the objective. Also the analysis allows doing this on different levels of abstraction which means that a top security manager can operate with collective elements (e.g., client's software) to locate the area of investments, while if it is needed the corresponding domain owner can easily expand the element to conduct the more fine-grained analysis.

Among the weaknesses of the approach one can identify rather informal way of eliciting security requirements. This will impose a lot on analyst, on his experience. On the other hand, using the enterprise model as a basis we make the proposal more objective then attack trees [24], [21], [19]. What we have is a *directed* refinement of security requirement which will help the analyst.

Identifying concrete threats for specific elements is also a quite subjective task but some semi-automation way can be also proposed here. Currently we are developing a pattern-based system which employs a catalogue of basic combinations of dependencies between model elements. After an additional analysis of context information about an element it should be possible to propose all (or more significant) threats from a predefined list that could be applied to the element. The list of threats is a compiled list taken from security guidelines and standards like [7], [13], [9] and mapped to basic and more complex patterns that are derived from our Enterprise Metamodel.

The most difficult part of the approach is identifying the weights. Although the correct identification of the values is a hard task we should take into account that the weights should be determined by the corresponding domain owners (e.g., weights showing possible impact on client's banking software are determined by the corresponding administrators) which have a good knowledge about the model elements. In other words, the approach helps easily distribute tasks among experienced actors (domain owners) because of its model-based nature rather than imposing whole responsibility on a high level security manager collecting, aggregating and analyzing low level details. The figures can be taken from the past experience of the domain owners. For example, knowing the past numbers of capturing control over the money server node and numbers of misuses of money transfer application one can easily deduce the relative impact. Then the analyst may use this number in the future.

VIII. Conclusion and future work

In the paper we have presented the approach which analyzes how well a business security objective is fulfilled. This approach helps domain owners to conduct a fine-graded assessment and find out the causes of failures (the weakest places in the security system). The proposed assessment is based on an enterprise modelling framework presented in [6], [12] that provides us a more objective way of constructing the Threat Graph (which can be seen as an attack tree).

As a future work we are going to apply the proposed approach in one of our projects to analyse the security of a distributed cross-institutional health data record network. One of the main focus in the study will be the evaluation of the complexity of the approach and its suitability for analysing service oriented architectures.

IX. Acknowledgement

Artsiom Yautsiukhin thanks European project EU-IST-IP-SERENITY and Bolzano-Innsbruck-Trento (BIT) school for supporting his internship in Innsbruck during which the work has been done.

References

[1] J.O. Aagedal, F. den Braber, T. Dimitrakos, B.A. Gran, D. Raptis, and K. Stolen. Model-based risk assessment to improve enterprise security. In *Enterprise Distributed Object Computing Conference, 2002. EDOC '02. Proceedings. Sixth International*, pages 51 – 62, Sept. 2002.

[2] Christopher J. Alberts and Audrey J. Dorofee. *Managing information security risks: the OCTAVE approach*. Pearson Education, 2002.

[3] Peter Bernus, Laszlo Nemes, and Günter Schmidt, editors. *Handbook on Enterprise Architecture*. Springer, 2003.

[4] S. Bistarelli, F. Fioravanti, and P. Peretti. Defense trees for economic evaluation of security investments. In *Proceedings of the First International Conference on Availability, Reliability and Security*, pages 416–423, Washington, DC, USA, 2006. IEEE Computer Society.

[5] F. den Braber, I. Hogganvik, M.S. Lund, K. Stølen, and F. Vraalsen. Model-based security analysis in seven steps – a guided tour to the CORAS method. *BT Technology Journal*, 25(1):101–117, 2007.

[6] R. Breu and F. Innerhofer-Oberperfler. Model based business driven IT security analysis. In *Proceedings of the Symposium on Requirements Engineering for Information Security (SREIS)*, August 2005.

[7] BSI (Federal Office for Information Security). IT Baseline Protection Manual. http://www.bsi.bund.de/english/gshb/manual/index.htm, 2004.

[8] S. A. Butler. Security attribute evaluation method: a cost-benefit approach. In *Proceedings of the 24th International Conference on Software Engineering (ICSE'02)*, pages 232–240. ACM Press, 2002.

[9] DCSSI, Direction centrale de la sécurité des systèmes d'information. EBIOS: Expression des Besoins et Identification des Objectifs de Sécurité. URL: http://ebios.cases-cc.org/, June 2005. Version 2.0.

[10] K. Clark, J. Dawkins, and J. Hale. Security risk metrics: Fusing enterprise objectives and vulnerabilities. In *Proceedings from the Sixth Annual IEEE Information Assurance Workshop on Systems, Man and Cybernetics*. IEEE Computer Society Press, 2005.

[11] L. A. Gordon and M. P. Loeb. *Managing Cybersecurity Resources: a Cost-Benefit Analysis*. McGraw Hill, 2006.

[12] F. Innerhofer-Oberperfler and R. Breu. Using an enterprise architecture for IT risk management. In *Proceedings of the ISSA 2006 Conference*, 2006.

[13] ISO (International Organization for Standardization). ISO/IEC 17799:2005 Information technology – Code of practice for information security management, 2005.

[14] S. Jha, O. Sheyner, and J. Wing. Two formal analys s of attack graphs. In *Proceedings of the 2002 IEEE Computer Society Security Foundations Workshop*, page 49, Washington, DC, USA, 2002. IEEE Computer Society.

[15] E. Johansson and P. Johnson. Assessment of enterprise information security - an architecture theory diagram definition. In *Proceedings of the 3nd Annual Conference on System Engineering Research*, March 2005.

[16] Jan Jürjens. Model-Based Security Engineering with UML. In A. Aldini, R. Gorrieri, and F. Martinelli, editors, *FOSAD 2004/2005*, volume 3655 of *Lecture Notes in Computer Science*, pages 42 – 77. Springer, Sep 2005.

[17] KPMG. *Fraud survey 2006*, 2006. available via http://www.kpmg.com.au/Portals/0/FraudSurvey\%2006\%20WP(web).pdf.

[18] Axel van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. In *5th IEEE International Symposium of Requirements Engineering*, pages 249 – 262, 2001.

[19] Van Lamsweerde Axel, Brohez Simon, De Landtsheer Renaud, and Janssens David. From System Goals to Intruder Anti-Goals: Attack Generation and Resolution for Security Requirements Engineering. In *Proceedings Workshop on Requirements for High Assurance Systems (RHAS'03)*, pages 49–56, Sept. 2003.

[20] B. B. Madan, K. Goseva-Popstojanova, K. Vaidyanathan, and K. S. Trivedi. A method for modeling and quantifying the security attributes of intrusion tolerant systems. *Performance evaluation journal*, 1-4(56):167–186, 2004.

[21] S. Mauw and M. Oostdijk. Foundations of attack trees. In *Proceedings of the 8th International Conference on Information Security and Cryptology*, Lecture Notes in Computer Science. Springer-Verlag, 2005.

[22] R. Ortalo, Y. Deswarte, and M. Kaaniche. Experimenting with quantitative evaluation tools for monitoring operational security. *IEEE Transactions on Software Engineering*, 25(5):633–650, 1999.

[23] R. Kelly Rainer, Thomas Marshall, Kenneth Knapp, and Gina Montgomery. Do Information Security Professionals and Business Managers View Information Security Issues Differently? *Information Systems Security*, 16(2):100–108, Mar 2007.

[24] B. Schneier. Attack trees: Modelling security threats. *Dr. Dobb's journal*, December 1999.

[25] O. Sheyner and J. Wing. Tools for generating and analysing attack graphs. In *Proceedings of Formal Methods for Components and Objects*, Lecture Notes in Computer Science. Springer-Verlag, 2005.

[26] Guttorm Sindre and Andreas L. Opdahl. Eliciting security requirements with misuse cases. *Requirements Engineering*, 10(1):34–44, January 2005.

[27] Basie von Solms and Rossouw von Solms. From information security to...business security? *Computers & Security*, 24(4):271–273, June 2005.

[28] SSE-CMM. *Systems Security Engineering Capability Maturity Model - SSE-CMM Model Document*. Carnegie Mellon University, version 3.0 edition, June 15 2003. available via http://www.sse-cmm.org/docs/ssecmmv3final.pdf.

[29] G. Stoneburner, A. Goguen, and A. Feringa. Risk management guide for information technology systems. Technical Report 800-30, 2001. available via http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf.

[30] Bomil Suh and Ingoo Han. The IS risk analysis based on a business model. *Information & Management*, 41(2):149–158, December 2003.

[31] L. Wang, A. Liu, and S. Jajodia. Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts. *Computer Communications*, 29(15):2917–2933, 2006.

[32] E. Zambon, D. Bolzoni, S. Etalle, and M. Salvato. A model supporting Business Continuity auditing & planning in Information Systems. Technical Report TR-CTIT-07-17, Centre for Telematics and Information Technology, University of Twente, Enschede, 2007.

[33] E. Zambon, D. Bolzoni, S. Etalle, and M. Salvato. Model-Based Mitigation of Availability Risks. Technical Report TR-CTIT-07-04, University of Twente, 2007.