A Property Based Security Risk Analysis Through Weighted Simulation

Timo Winkelvos, Carsten Rudolph, Jürgen Repp
Fraunhofer Institute for Secure Information Technology, SIT
Rheinstraße 75, 64295 Darmstadt, Germany
Email: timo.winkelvos, carsten.rudolph, juergen.repp@sit.fraunhofer.de

***Abstract*—The estimation of security risks in complex information and communication technology systems is an essential part of risk management processes. A proper computation of risks requires a good knowledge about the probability distributions of different upcoming events or behaviours. Usually, technical risk assessment in Information Technology (IT) systems is concerned with threats to specific assets. However, for many scenarios it can be useful to consider the risk of the violation of particular security properties. The set of suitable qualities comprises authenticity of messages or non-repudiability of actions within the system but also more general security properties like confidentiality of data. Furthermore, as current automatic security analysis tools are mostly confined to a technical point of view and thereby missing implications on an application or process level, it is of value to facilitate a broader view including the relation between actions within the IT system and their external influence. The property based approach aims to help assessing risks in a process-oriented or service level view of a system and also to derive a more detailed estimation on a technical level.**

Moreover, as systems' complexities are growing, it becomes less feasible to calculate the probability of all patterns of a system's behaviour. Thus, a model based simulation of the system is advantageous in combination with a focus on precisely defined security properties.

This paper introduces the first results supporting a simulation based risk analysis tool that enables a security property oriented view of risk. The developed tool is based on an existing formal validation, verification and simulation tool, the Simple Homomorphism Verification Tool (SHVT). The new simulation software provides a graphical interface for a monitor automaton which facilitates the explicit definition of security properties to be investigated during the simulation cycles. Furthermore, in order to model different likelihoods of actions in a system, weighting factors can be used to sway the behaviour where the occurrence of events is not evenly distributed. These factors provide a scheme for weighting classes of transitions. Therefore, the tool facilitates probabilistic simulation, providing information about the probability distribution of satisfaction or violation of specified properties.

I. Introduction

There are a lot of different interpretations of the term *risk*. People use it to name possible negative events, the consequences thereof or the probabilities, frequencies and expected damages of any incidents. Inherent to all of these is an uncertainty of the occurrence of particular events as well as the outcome. Thus, some estimation on the behaviour of systems is inevitable, when analysing risk. It usually involves identification of vulnerabilities and threats, estimating the probabilities of the occurrence of events and cost damage of successful attacks. In addition, one has to estimate costs and efficiency of countermeasures.

The term *risk* as it is used with respect to the *risk analysis* approach of this paper solely refers to the *probability of a violation of security properties*. While additional measures of the impact would be needed for a general evaluation of risk, this interpretation makes sense here, as the violation of a security property could lead to various incidents and thus to large range of different impacts. For example, the effect of a violation of the confidentiality of some data depends not only on the value of the data, but also on the way an attacker capitalises on the information, which is out of scope of this paper.

Model-based approaches and automated simulation have been used in the past for various steps of security analysis processes. They can help to determine the feasibility for particular attacks and to optimise the use of countermeasures against these attacks or combinations thereof. However, the total risk of any security issues in Information and Communication Technology (ICT) systems cannot be easily determined from risks for particular types of attacks. Most relevant is the relation between actions within the ICT system and the actual impact on the organisational or business level. Determination of effects and financial losses can only be done from a perspective that includes the environment and requirements. Usually, the relation between an issue on ICT level and e.g. the effect on business processes and assets is done manually and not sufficiently supported by model-based approaches. Rather than looking at particular weaknesses it would make more sense to estimate risks relative to particular security requirements concerning the entire system. A useful metrics for security risk can be based on probability distributions for the violation of sets of security properties.

The need for quantifiable security is strong [1], and the search for solutions is not new [2] but still lacks sufficient solutions [3]. While a quantified forecast of the level of secu-

rity is needed, different factors make this task very hard [4]. First, there is a lack of consistent statistical data to learn from and to create a firm statistical regression as it is done in other research areas. Second, even if statistical data were available, significance of the analysis would be decreased because of the systems' enormous heterogeneity. The technological environments quickly change and complicate the deduction of security quantification [5], [6]. Furthermore, upcoming vulnerabilities and exploits are unknown and attacker's knowledge and intents are not directly reflected in the correct system behaviour. Thus, a logical first step of a security risk analysis is to create a model of the system that makes use of any available data about the subject, is detailed enough to be used for (high-level) security requirements specifications, abstracts from as much unneeded information as possible and that allows the analyser to control the information sources the model builds upon. Refined versions of such a model can support different related layers of security risk analysis each considering issues on the particular layers.

This paper introduces ongoing research on risk identification and quantification in the field of IT security. Advantages of model checking have been shown with the SHVT ([7]) in several security and reliability related contexts. In those works properties are proven or else counterexamples are found. Rather than aiming at such discrete results our current goal is to find measures to quantify uncertainty inherent to information systems. These quantified results are supposed to be integrated into, and thereby facilitate the risk assessment process of information security risk management methods.

The additional functionality of a simulator is introduced to the SHVT enabling to run (or simulate) a system model repeatedly. An additional automaton, called a Monitor, facilitates the modelling of certain aspects of the systems' behaviour and monitor such properties during the run of the simulation. Furthermore, in order to model different likelihoods of actions in a system, weighting factors can be used to model a behaviour where the occurrence of events is not evenly distributed. For each single simulation cycle, the tool can calculate and store various information. Most importantly, it keeps track of whether properties described by the Monitor are satisfied or not. With a high number of simulation cycles, this information can be used to calculate the probability distribution of the satisfaction or violation of monitored properties. Thus, the extended features enable to simulate and concurrently monitor and investigate characteristics that are of importance with respect to certain issues of interest. The results are used to elicit a means of forecasting of security issues and therefore to calculate probabilities of risks.

The simulation approach is useful in cases of complex systems. Calculating every possible state of a system and eliciting a probability distribution for the possible actions in each of the states is not feasible for complex real world scenarios. The approach of simulating facilitates two different aspects of optimisation. On the one hand, the problem of irrelevant characteristics is tackled through the use of a monitor and thus through the concentration on specific properties. This greatly reduces the amount of data stored during the analysis. On the other hand, the infeasibility of calculating the probability of all states of a reachability graph is addressed by the simulation of which any cycle is equivalent to only one path through the transition system.

In the following section a brief survey of related scientific work is given before section III describes the modelling approach, the simulation of system behaviour and the interplay with the monitor. Section IV reviews the work and illustrates future work followed by the conclusion in section V.

7th July 2011

II. Related Work

Today, the quantification of IT risks is mostly based on manual inspection (check lists) and subjective evaluation of discrete risks in accordance to structured methods like the ISO standard 27005 [8], CORAS [9], [10] or OCTAVE [11]. While these techniques make sense in that they allow to structure the risk analysis and management and evaluate single risks in a relatively consistent way, they have some disadvantages. They have a very high subjective influence, lack automated search capabilities and therefore they are prone to errors and very complex and costly [12]. This work aims to be complementary to the above mentioned methods by introducing tool based support of the risk assessment process where possible.

A lot of approaches use graph representations to support the understanding of security risks [13]–[15]. Wang et al. [16] introduced an attack graph based approach to investigate the likelihood of attacks. Another recent work combines attack trees with Markov processes [17]. Noel et al. [18] insert a variety of information gathered by life-systems into their otherwise static attack graph based topological vulnerability analysis. While such works give static probabilities to events representing e.g. actions the attacker can conduct, in our approach an action's probability is dynamically calculated according to the state of the systems and thus depending on previous actions and alternative events. Attack Graphs and similar attack modelling schemes have been used to analyse computer attacks since the 1990's e.g. by Dacier [2], Phillips and Swiler [19] and Gorodetski and Kotenko [20]. A survey of attack graphs and security analysis is given in [21].

For handling uncertainties in information security risk analysis usually statistical distributions according to the different subjects, requirements and assumptions are applied [2], [17], [22]–[24]. Using such distribution functions leads to an abstraction of the source of randomness and complexity from internal factors. In combination with a complex, state based model, such as is used in this paper, this forces a less flexible input and disallows to differentiate in the details. In particular, the effect of small changes inside of the system (e.g. an additional security mechanism) on the stochastic results of the simulation would hardly be analysable. In order to create detailed (stochastic) results, one needs to find detailed input, which is why the relative weighting factors have been chosen. A fitting distribution function could be used to add external randomness (like a Weibull distribution to estimate the time to

failure of some equipment or a normal distribution for error rates [25]).

Several different works introduce model checking in conjunction with attack graphs and security analysis, respectively. Ritchey and Ammann [26] use the model checker SMV (http://www.cs.cmu.edu/~modelcheck/smv.html) to compute counterexamples which reflect possible malicious behaviour. A similar approach by Sheyner et al. [27] automatically generates attack graphs and enables reliability analysis. Their proposal of forecasting behaviour is based on the assumption of some statistical information. The input to the model of Phillips and Swiler [19] consists of attacker profiles and attack templates and they compute a reachability graph thereof to analyse the level of security using shortest path algorithms. Another example for attack graph generation through model checking is given by Wing [28]. Related is the work by Rieke [29] which adds a variety of analysis tools and evaluation algorithms based on the reachability graph. Mehta et al. [30] point out that attack graphs can exceed controllable size and therefore propose to rank attack graphs and states thereof and introduce a probability metric. Thus, they try to restrict the analysis to the most important or most endangered aspects of a system.

An interesting area of research is the field of probabilistic model checking. This is about analysing systems that have an intrinsic stochastic behaviour and propagating the known probabilities through a complex model [31]. However, the proposed method of this paper approximates the uncertain future behaviour with simulation, resulting in probabilities. A probabilistic model checking analysis offers information like: a statement will be true *at least or at most with probability p*. The behaviour of large information systems with a lot of possibly unknown input can be considered as stochastic, so this field may provide insights into the elicitation of the weighting factors.

III. Simulation & Monitoring

The goal of this work is to support IT risk assessment processes. The main approach is to dynamically and randomly simulate a system's behaviour with on-the-fly verification of properties represented by a monitor automaton. The probabilities of state transitions in the simulation are dynamically derived from weighting factors added to the model in advance. Weighting factors can be associated with the transitions (e.g. events or actions) of the model that allow to cope for different likelihoods of events. This approach has been implemented as extension to the SHVT. As this paper aims on the description of the approach, we use a scenario that has been used and published before. Here, we confine ourselves to describing those aspects of the scenario, that are necessary for an adequate understanding.

This section introduces the elements of the approach and explains the mode of operation of the simulation with the help of an example. Figure 1 shows the relationship between the model, the simulation and the monitor.

A. The Model

First, the object of investigation has to be modelled. In the case of the example, the object is a networking system comprising multiple hosts in different network zones, a set of access policies depending on these zones and a set of vulnerabilities (see the example in subsection III-D). The latter can be used for exploits by an attacker that wants to get access to a machine of the network. The model has to represent every element of the configuration that has an influence on possible events and the options of actions of the modelled roles (e.g. the attacker). Thus any state of the modelled system reflects the configuration of the network and the condition and properties of every instance like, for example, the access rights of an attacker w.r.t. a specific server. Note that the tool is not focused on a particular type of applications (e.g. IT networks). It can be easily applied to totally different scenarios such as protocol specifications, business processes or service level scenarios.

In order to simulate a system, it has to be formally represented in a model where the dynamic behaviour of the system is defined by a labelled transition system (LTS) that is usually computed from the specification by a tool. An LTS consists of a set of *states* $\mathcal{Q}$, *actions* $\mathcal{R}$ and a set of *transition relations* $\mathcal{Q}\times\mathcal{R}\times\mathcal{Q}$. The SHVT can be combined with any specification tool generating labelled transition systems. The current implementation of SHVT uses Asynchronous Product Automata (APA) in order to model systems under investigation. APA are a flexible operational specification concept of cooperating systems [7], [32]. An APA consists of a family of states, a family of elementary automata and a family of neighbourhood relations ($\mathcal{T}$). The latter can be interpreted as the set of all transitions (e.g. actions or events) that change the states of the APA. They are of the form $(s, t, s_{next}) \in \mathcal{T}$, where s and s_{next} are consecutive states and t the corresponding transition. Thus, the behaviour of the system is described by the sequences of state transitions. Transition patterns optionally include relative weighting factors to dynamically derive random choices in the simulation. The executed model (i.e. the reachability graph for the specified system) can be seen as an LTS. One cycle of the simulation computes a single path through this LTS.

B. The Simulation

Any run of the simulation consists of cycles which start with an initial state and step from one state of the model to the next until one of different restart conditions is met. The sum of these cycles should create a meaningful sample.

In any state of the (modelled) system, there might be different transition alternatives. The simulation's traversal differs from that of a full reachability analysis in that only one transition is selected to be triggered next. In every state of a simulation one transition is randomly chosen from the set of possible transitions. This choice is swayed by weighting factors that are associated with the transition patterns of the model.

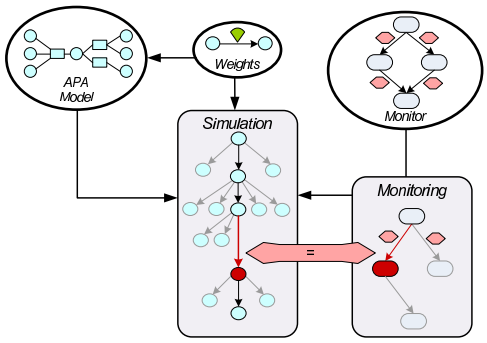

Figure 1. Relationship between model, simulation and monitor

Let $\mathcal{T}_x$ be the set of the n possible transitions $t_i, i = 1, \ldots, n$ in a particular state s_x of the simulation and let g_i be the weight of the respective transition t_i. Then the likelihood of transition $t_{next} \in \mathcal{T}_x$ being the next transition to be triggered in the state s_x is defined by the state dependent probability distribution $\mu_{s_x} : \mathcal{T}_x \to [0, 1]$ with

$$\mu_{s_x}(t_{next}) = \mu_{s_x}(g_{next}) = \frac{g_{next}}{\sum_i g_i}, \tag{1}$$

where $\sum_i \mu_{s_x}(t_i) = 1, \forall s_x$.

Thus, the actual probability of any event in a given state of the model depends on it's own weight as well as on the respective state and the sum of weights of the alternatives. It is also possible to have weights dynamically changing with the behaviour of the system.

A cycle of the simulation ends when either there are no subsequent transition alternatives (a dead state is reached), or a defined maximum number of steps per cycle is reached, or a new cycle is triggered by the Monitor as described below. The next cycle starts again with the initial state. The run of the simulation either ends after a predefined maximum number of steps or is stopped manually.

C. The Monitor

As illustrated in Figure 1, during the run of the simulation, one or several automata monitor the resulting behaviour. A *monitoring automaton* or monitor consists of

- a set $\mathcal{M}$ of *labelled states* m,
- an alphabet Λ of *predicates* λ and
- a *transition relation* $\mathcal{T}_\mathcal{M} \subseteq \mathcal{M} \times \Lambda \times \mathcal{M}$.

One or more of a monitor's states are initial states m_0 assigned to the initial state of the LTS under investigation.

Each predicate $\lambda \in \Lambda$ of the Monitor is of the form $([pre], [tn][ivp], [suc])$ defining checks to transitions of the LTS w.r.t. predecessor (pre), transition (name tn, interpretation variables ivp) and successor state (suc). The predicates of the monitor are positive if all three $lambda$-terms are positive or empty. Each $\lambda \in \Lambda$ is associated with one of the transitions $\mathcal{T}_\mathcal{M}$ of the monitor automaton.

The monitor has been developed to allow for queries of different characteristics of the states and state transitions of an LTS. It can reflect attributes concerning all parameters of LTS state transition. By allowing more than one transition of the monitor to be triggered at the same time, different properties can be monitored within one simulation. Thus, the overall state of the monitor can contain several *active* states. Edges leading from the active states are considered as *active*, too. The transition relation defines the active edges for all active states. During the run of the simulation, the active elements of the monitor are assigned to the current state of the LTS (as part of the respective state components). Whenever a transition of the model is chosen to be the next step of the simulation, all predicates of the monitor's active edges are evaluated.

Let $\mathcal{A} \subseteq \mathcal{M}$ be the set of n active states $m_a, a = 1 \ldots n$ of the monitor. Let the sets of adjacent transitions' predicates be Λ_a, with $\lambda_{a_i} \in \Lambda_a$, $i = 1 \ldots k$ for the k active transitions of m_a and $\Lambda_a \subseteq \Lambda$. The set of transitions from the active states are therefore $(m_a, \lambda_{a_i}, m_{a_i})$. If a model's state transition $(s, t, s_{next}) \in \mathcal{T}$ is triggered during the simulation, then for all active states m_a the predicates λ_{a_i} are asserted according to the following equation.

$$\begin{aligned} \lambda_{a_i}(s, t, s_{next}) &= true \Leftrightarrow \\ pre(\lambda_{a_i})(s) = true &\wedge tn(\lambda_{a_i}) = name(t) \wedge \\ ivp(\lambda_{a_i})(t) = true &\wedge suc(\lambda_{a_i})(s_{next}) = true \end{aligned} \tag{2}$$

For any positive assertions the respective subsequent states are activated (marked in Figure 1) and the previously active state of the monitor is deactivated after all active edges have been evaluated. All transitions of the monitor automaton are recorded.

The output of a simulation run comprises statistics on the number of steps and cycles of the simulation, the total number of state transitions of the monitor and statistics about the reasons of restarts of simulation cycles. Furthermore, for each of the monitor's states, a statistics on the frequency and duration (number of simulation steps) of activation is given as well as the number of cycle-restarts while the respective state is activated.

D. Simulating an Example Scenario

The following brief example shows one possible application of the weighted simulation and the monitor automaton. In order to make it short, the description concentrates on relevant aspects of a scenario published before by Rieke [29]. Please note that network intrusion is but an illustration and not the sole or main focus of this work.

As mentioned above, the scenario is an ICT Infrastructure comprising sets of hosts in different network zones, a database server, an intrusion detection system and one ore more attackers. A security policy defines which privileges are needed to perform any activities, in particular concerning access to the database server. Further, hosts can be prone to a set of vulnerabilities. The model reflects this scenario through roles that can perform actions, modelled as state transitions. The attacker can now try to exploit vulnerabilities with the goal of getting access to the database server (see also Figure 2).

The output of one simulation is shown in table II. In the first column, one can find the states of the example monitor

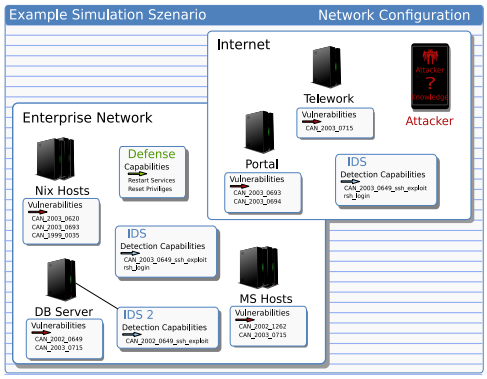

Figure 2. Scenario network overview

Table I
STATS OF THE SIMULATION EXAMPLE

14285	simulation cycles
100000	steps (in Model)
42412	transitions (of Monitor)

Table II
RESULTS OF EXAMPLE MONITOR

State of Monitor	Steps of Sim.	Transitions
Initial	55560	0
MultipleAccess	3900	2023
RootOnNixHost	31068	10874
UnexpectedBreach	0	0
UserOnMsHost	6400	2726
BreachMultiple	290	193
BreachViaMsHost	644	269
BreachViaNixHost	2139	1291
Sum of Breaches	3073	1753
Ainitial2	62470	0
AAllRootOnNixHost	37531	12518
ARshLoginToNixHost	0	0
AmanDBToNixHost	13	11
AsshExpStealthToNixHost	11368	3856
AsshExpToNixHost	26150	8651

automaton. Column 2 shows the number of steps of the simulation during which the monitor stayed in the respective state. The rightmost column shows how often, during the simulation, the respective state was reached or in other words, how many of the simulation's cycles lead to a system state with the respective properties.

For statistics on the simulation see table I. The aim of the simulation in this example is to find and compare successful attack strategies. The monitor automaton's states reflect the known preconditions of an event that leads to unauthorised access privileges for the attacker. The automaton checks which of the preconditions are fulfilled whenever the attacker gets privileges on the database server. Furthermore, in order to avoid misinterpretations a state of the monitor reflects simultaneous fulfilment of both preconditions and a transition is added that allows to decide whether a breach is possible without any of the mentioned privileges, e.g. that has not been thought of before (see the monitor's state *UnexpectedBreach* in table II).

By changing the monitor automaton the focus of the simulation can be refined easily. A more complex design of the monitor allows for a more detailed forecast of system behaviour and the coherences within the system. The example monitor has been enhanced in order to evaluate the attributes during the breach and, at the same time, the exploits the attacker most likely uses to get the respective privileges. The respective findings can be found below the double line in table II. This run of the simulation comprises 100000 steps in 14285 cycles. In these 1753 breaches (successful attacks) occurred. Of these 1484 occurred after the attacker obtained privileges on the *nix_host* which equals 84.7%. Thus it is easy to elicit the advantage of a concentration of efforts on this host in order to decrease the risk.

As the Unix Host has been identified to be critical, the enhanced monitor was created in order to analyse the activities leading to a compromise of the critical host. Simulating the model with the enhanced monitor results in the finding, that in only 11 of 14285 cycles of a simulation the exploit of a man-db vulnerability (CVE-2003-0620) was used to get root privileges on the critical host, while more than 99 per cent of successful breaches are based on exploits of an openssh vulnerability (CVE-2003-0693).

The example shows how to iteratively examine expected system behaviour with small effort using the simulation and the monitor automaton. The likelihood of different events or system traces can be statistically determined. However, the example also shows, that a simulation can only give answers to questions that go beyond the model, because it has to incorporate the respective aspects on an adequate level of abstraction. For that reason, future work will focus on creating interfaces to tools like Security Information and Event Management (SIEM) appliances in order to automatically generate parts of the model and in particular the weights of transitions.

IV. DISCUSSION AND FUTURE WORK

The monitor is a powerful tool to strictly define those patterns of system behaviour, that have an implication on properties under investigation. In the example of III-D, these were mostly privileges and information on how they are gained. System traces that indicate security properties beyond access control can be monitored as well. In other use cases of the ongoing work, the monitor is used to define complex properties with regard to the interplay of different contexts, like the application state, network configuration, physical properties and different processes interlinked with the former (e.g. a maintenance process in a critical infrastructure).

With the Approach presented here, it is obvious that an action's likelihood depends not only on a weight classification

defined in advance but also reflects the corresponding state of the system. Deriving the probability distribution over the alternatives in a given state of the model from classes of weights enables to analyse larger, more heterogeneous systems than with normal probabilistic approaches.

Simulation of the APA model also reduces the number of actions processed in comparison to evaluating all possible state transitions in a model's reachable state space. Furthermore the generation of likelihood distributions only for those states reached during a simulation is feasible for large models. This allows for a dynamical computation of the probabilities of a state's successors and thus enables an assessment of large systems. Some further aspects of the ongoing work are discussed in the following.

A. Model

Creating a model of a system under investigation always idealises reality. Nevertheless, it is a means to a) abstract from insignificant complexity of the reality, b) capture the essence of an issue under investigation and c) automatically evaluate an abstraction of reality using a computer. Whenever a model is created, the model and those working with the model have to regard any assumptions made and the aspects abstracted from. Modelling vulnerabilities, exploits and attacker's capabilities is not the only way of analysing a system's security issues. Instead, an advantage of the approach is the opportunity to simulate various different kinds of LTS models. These can describe diverse systems on different levels of abstraction and thereby lay the focus on different (security) attributes of a system. The mentioned diversity could include cyber-physical models, work-flow specifications and business process information. Furthermore, the combination of the APA model with the very flexible monitor automaton allows for fast succession of analysis regarding different issues.

Otherwise, modelling of systems is a science in itself or sometimes even called an art [25]. It certainly is a complex task, so further work will be about finding out what could possibly be modelled with what effort and how to find constraints. An interesting aspect of future investigation is whether it is possible to create templates for APA models (or any other, possibly simpler model that can produce an LTS) describing the object of investigation on the one hand and different suitable monitors for predefined properties on the other. Especially the latter seems to be promising, as most of the security properties are of general interest and the corresponding indicators can be found in different systems. Besides, the automation of model creation from existing data is already and will be further examined.

B. Weighting Factors

According to the status quo in information security risk assessment literature (see section II), likelihood quantification is a very tedious task. Either, probabilities of risks are gathered by working through catalogues and assessing every single risk, or all actions that lead to negative incidents are evaluated separately. Both ways seem more demanding and error prone than the definition of weights for classes of actions in accordance to each other, as proposed in this paper. Moreover, with the proposed method, the number of single evaluations is much smaller.

The determination of the weights can be integrated into workshops for threat and vulnerability identification that are part of some risk assessment methods, e.g. OCTAVE [11]. Such workshops are held by security experts, decision makers and personal that know the system under investigation best, which makes these groups the perfect team to evaluate the relative weights of transitions (e.g. events) in the model of the system. Also, probabilities identified during such workshops today, could be used as input without changing the standard process.

Furthermore, parameters such as cost of an attack, expertise required, availability of exploits, or financial gain for the attacker can be considered in computing the weight factors. Additionally, it has been proposed that potential attackers would always choose those attack vectors that promise to minimise the demand w.r.t. programming know how and time to success [2]. Time of such processes might be measurable and the know how could be rated and thereby provide a basis to elicit weights.

Nevertheless, a concentration of future work will be aimed at reducing the subjective factor and introducing automation concerning the creation of the weighting factors. This first approach is based on a static predefinition of weights. As described in the example above it makes sense in many cases to define dynamic weights that are computed as a function over the current state of the LTS, the history of the simulation run or the monitoring automaton. Also attack strategies can be reflected in dynamic weights. Then, if a particular pattern occurs in the simulation, weights are swayed for all steps being part of this attack strategy.

It is also planned to facilitate dynamic adjustment of the weights through linking to SIEM solutions. Those systems aggregate, correlate and analyse security-related data in live systems. Exploring the data generated by SIEM systems in order to define and influence the weights could be beneficial in two modes of implementation. Either historical SIEM data is used to derive the weights for a simulation, or the simulation would run constantly and parallel to a SIEM solution and the live events change the weights during the run of the simulation.

C. Simulation

Through the introduction of a weighted simulation, the model checking approach, that is intrinsically analytic and (concerning the result) deterministic has been extended to include stochastic capabilities. This enables to make forecasts w.r.t. the likelihood of certain traces of action and thereby approximate unknown aspects of future behaviour.

In a simulation or a similar quantitative security analysis in the field of information security, it is common to use probability distribution functions to model unknown behaviour and cope with uncertainties. The variables are chosen to fit the situation as good as possible. Mostly, exponential

distributions are used to model attackers behaviour but other distributions have been applied to security analysis, too [17], [22]–[24]. Defining probabilities according to such functions unlinked to system intrinsic properties carries the risk of abstracting from system properties that should have been investigated. In this paper, probabilities of events are generated through the random choice of active transitions and therefore a statistical distribution function is not used. Nevertheless, it might be worthwhile to consider a combined approach if for a large number of events a fixed statistical distribution correctly models the expected occurrence.

Future work will include an analysis of the statistical properties of the simulation which will help calibrating the parameters within the process. One approach will be a comparison between the simulation results and those of a numerical likelihood calculation based on a reachability analysis using the same set of weights. A statistical analysis of the results will be conducted in order to elicit the conditions of a meaningful probe.

V. Conclusion

The principal goal of the ongoing work presented here is to assist the information security risk management process through the development of a model based methodology of risk assessment. Our approach is to simulate a system that is modelled as an automaton based LTS. The simulation is monitored by an additional automaton that reflects security properties and enables to analyse the system behaviour regarding these properties. Weighting factors are added to classify transitions, relatively swaying the simulation. These components have been successfully implemented.

The property based modelling allows to simulate the system on various levels of abstractions. It even enables analysis of unforeseen effects that is disregarded by risk assessment methods that focus on asset violation, because well-defined security properties do not change with the system set up and thus enable to reuse (branches of) the monitor. Furthermore, the weighted simulation facilitates flexible evolution of the analysis through the successive adjustment of parameters or model.

Compared to other approaches presented in this paper the relative weighting of classes of transitions is less complex in terms of quality and quantity. This distinguishes the approach from manual likelihood estimation of any single event of the system behaviour or from evaluation of risks according to threat catalogues or asset oriented risk estimation. Furthermore, as the weighting factors will be used to interface the simulation with system and security events from SIEM solutions, they are a key factor of the integration of the approach with existing security management software and thus of the automation and practical usability of the risk analysis.

The model based simulation is an effective way of making forecasts through approximating probabilities of future behaviour. The approach is very flexible, allowing different ways to adjust parameters successively. Future work will aim at automation, development of the forecasting algorithm and the minimisation of subjective factors through connection to real measurable data.

List of Acronyms

SHVT	Simple Homomorphism Verification Tool
ICT	Information and Communication Technology
IT	Information Technology
SIEM	Security Information and Event Management
LTS	labelled transition system
APA	Asynchronous Product Automata

List of Symbols

$\mathcal{Q}$	The set of states in an LTS.
$\mathcal{T}$	The set of transitions of the APA.
s_i	A particular state of the LTS, indexed i.
$\mathcal{T}_x$	The set of transitions alternatives in a state s_x.
t	Any transition.
t_i	A particular transition, indexed i.
g_i	The weight of the transition t_i.
μ	The probability distribution function
$\mathcal{M}$	The set of states of the Monitor.
m	a particular state of the Monitor.
$\mathcal{A}$	The set of active states of the Monitor.
λ	A predicate of the Monitor.
Λ	The alphabet of predicates of the Monitor.
pre	Predicate regarding the predecessor state.
tn	Predicate regarding the name of the transition.
ivp	Predicate regarding the interpretation variables of the transition.
suc	Predicate regarding the successor state.

Acknowledgment

This paper was written as part of the MASSIF project (http://www.massif-project.eu/), which is funded by the European Commission, and as part of the CASED project (http://www.cased.de), funded by the German Hessian government.

References

[1] D. Geer Jr, K. Hoo, and A. Jaquith, "Information security: why the future belongs to the quants," *IEEE Security & Privacy*, vol. 1, no. 4, pp. 24–32, 2003.

[2] M. Dacier, Y. Deswarte, and M. Kaaniche, "Quantitative assessment of operational security: Models and tools," *LAAS Research Report*, vol. 96493, 1996.

[3] V. Verendel, "Quantified security is a weak hypothesis," in *Proc. of NSPW09*, 2009.

[4] J. McHugh, "Quality of protection: measuring the unmeasurable?" in *Proc. of the 2nd ACM workshop on Quality of protection*. ACM, NY, USA, 2006, pp. 1–2.

[5] A. Jaquith, *Security metrics: replacing fear, uncertainty, and doubt*. Addison-Wesley Professional, 2007.

[6] S. Schechter, "Computer security strength & risk: A quantitative approach," Ph.D. dissertation, Harvard University Cambridge, Massachusetts, 2004.

[7] P. Ochsenschläger, J. Repp, and R. Rieke, "The SH-verification tool," in *Proc. 13th International FLorida Artificial Intelligence Research Society Conference (FLAIRS-2000)*, 2000, pp. 18–22.

[8] ISO/IEC, *ISO/IEC 27005, Information technology - Security techniques - Information security risk management*. ISO/IEC, 2008.

[9] R. Fredriksen, M. Kristiansen, B. Gran, K. Stolen, T. Opperud, and T. Dimitrakos, "The CORAS framework for a model-based risk management process," *Lecture notes in computer science*, pp. 94–105, 2002.

[10] J. Aagedal, F. den Braber, T. Dimitrakos, B. Gran, D. Raptis, K. Stølen *et al.*, "Model-based risk assessment to improve enterprise security," in *Proc. of 6th International Conference on Enterprise Distributed Object Computing (EDOC02)*, 2002.

[11] C. Alberts and A. Dorofee, *Managing information security risks: the OCTAVE approach*. Addison-Wesley Professional, 2002.

[12] K. Buyens, B. De Win, and W. Joosen, "Empirical and statistical analysis of risk analysis-driven techniques for threat management," in *Proceedings of the The Second International Conference on Availability, Reliability and Security*. IEEE Computer Society, 2007, pp. 1034–1041.

[13] D. Balzarotti, M. Monga, and S. Sicari, "Assessing the risk of using vulnerable components," in *Proceedings of the 2nd ACM workshop on Quality of protection*. Springer, 2005.

[14] M. Benini and S. Sicari, "A mathematical framework for risk assessment," *New Technologies, Mobility and Security, Signals and Communication, Springer-Verlag*, pp. 459–469, 2007.

[15] M. Sahinoglu, "Quantitative risk assessment for dependent vulnerabilities," in *Reliability and Maintainability Symposium, 2006.*, 2006, pp. 82–85.

[16] L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia, "An attack graph-based probabilistic security metric," *LNCS*, vol. 5094, pp. 283–296, 2008.

[17] L. Piètre-Cambacédès and M. Bouissou, "Beyond attack trees: dynamic security modeling with boolean logic driven markov processes (bdmp)," in *2010 European Dependable Computing Conference*, 2010.

[18] S. Noel, M. Elder, S. Jajodia, P. Kalapa, S. OHare, and K. Prole, "Advances in Topological Vulnerability Analysis," 2009.

[19] C. Phillips and L. Swiler, "A Graph Based System for Network Vulnerability Analysis," in *Proc. of NSPW1998*. ACM, 1998, pp. 71–79.

[20] V. Gorodetski and I. Kotenko, "Attacks against computer network: Formal grammar-based framework and simulation tool," in *Recent Advances in Intrusion Detection*. Springer, 2002, pp. 219–238.

[21] R. Lippmann and K. Ingols, "Annotated Review of Past Papers on Attack Graphs," Tech. Rep., 2005.

[22] B. Littlewood, S. Brocklehurst, N. Fenton, P. Mellor, S. Page, D. Wright, J. Dobson, J. McDermid, and D. Gollmann, "Towards operational measures of computer security," *Journal of Computer Security*, vol. 2, no. 2-3, pp. 211–230, 1993.

[23] E. Jonsson and T. Olovsson, "A quantitative model of the security intrusion process based onattacker behavior," *IEEE Transactions on Software Engineering*, vol. 23, no. 4, pp. 235–245, 1997.

[24] M. McQueen, W. Boyer, T. McQueen, and S. McBride, "Empirical Estimates of 0Day Vulnerabilities in Control Systems," in *Proceedings of the SCADA Security Scientific Symposium*, 2009.

[25] A. M. Law, *Simulation modeling & analysis*, fourth edition ed. McGraw-Hill, 2006.

[26] R. Ritchey, P. Ammann, A. Booz, H. Inc, and F. Church, "Using model checking to analyze network vulnerabilities," in *2000 IEEE Symposium on Security and Privacy, 2000. S&P 2000. Proceedings*, 2000, pp. 156–165.

[27] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing, "Automated generation and analysis of attack graphs," in *2002 IEEE Symposium on Security and Privacy, 2002. Proceedings*, 2002, pp. 273–284.

[28] J. Wing, "Scenario graphs applied to network security," *Information assurance: dependability and security in networked systems*, 2008.

[29] R. Rieke, "Modelling and analysing network security policies in a given vulnerability setting," *Critical Information Infrastructures Security*, pp. 67–78, 2006.

[30] V. Mehta, C. Bartzis, H. Zhu, E. Clarke, and J. Wing, "Ranking attack graphs," in *Recent Advances in Intrusion Detection*. Springer, 2006, pp. 127–144.

[31] M. Z. Kwiatkowska, G. Norman, and D. Parker, "Quantitative analysis with the probabilistic model checker prism," *Electronic Notes in Theoretical Computer Science*, vol. 153(2), pp. 5–31, 2005.

[32] S. Gürgens, P. Ochsenschläger, and C. Rudolph, "Role based specification and security analysis of cryptographic protocols using asynchronous product automata," in *Database and Expert Systems Applications, 2002. Proceedings. 13th International Workshop on*, 2002, pp. 473–479.