Externalizing Behaviour for Analysing System Models

Marieta Georgieva Ivanova[1]*, Christian W. Probst[1], René Rydhof Hansen[2], and Florian Kammüller[3]
[1] Technical University of Denmark, Denmark
{mgiv, cwpr}@dtu.dk
[2] Aalborg University, Denmark
rrh@cs.aau.dk
[3] Middlesex University, UK
F.Kammueller@mdx.ac.uk

Abstract

System models have recently been introduced to model organisations and evaluate their vulnerability to threats and especially insider threats. Especially for the latter these models are very suitable, since insiders can be assumed to have more knowledge about the attacked organisation than outside attackers. Therefore, many attacks are considerably easier to be performed for insiders than for outsiders. However, current models do not support explicit specification of different behaviours. Instead, behaviour is deeply embedded in the analyses supported by the models, meaning that it is a complex, if not impossible task to change behaviours. Especially when considering social engineering or the human factor in general, the ability to use different kinds of behaviours is essential. In this work we present an approach to make the behaviour a separate component in system models, and explore how to integrate in existing models.

Keywords: system models, static analysis, human behaviour

1 Introduction

Insider threats pose difficult problems for organisations. By their very nature, insiders have access to and knowledge about the organisation's infrastructure, data, and workflows, and they use these in their everyday work to fullfil tasks. Determining whether a certain action by an insider is legitimate (regular work) or illegitimate (insider attack), is close to impossible because the insider's purpose for performing the action is not observable. This problem exists both if the action involves assets that the insider is not allowed to access, and even more so, if the access to the asset is allowed.

Organisations often distinguish between threats originating from the inside and the outside. While this in principle makes sense (and many organisations are much better prepared against attacks from the outside than against those from the inside), it is also risky. If an outsider is able to get an insider to perform a certain action, the insider becomes, voluntarily or involuntarily, part of the attack. Social engineering is a typical kind of attack used in these scenarios [16]; it aims at making an insider perform an action that he is allowed to perform, were it part of his daily work.

For various reasons, neither regulating insider actions nor surveilling them are viable options. Uber-regulation easily results in disgruntled employees, and the human mind can be ingenious when having to circumvent security precautions and policies. Uber-surveillance, on the other hand, is illegal in many parts of the world, and even if it is admissible, it is unclear how to draw meaningful conclusions from

Journal of Internet Services and Information Security (JISIS), volume: 3, number: 3/4, pp. 52-62

*Corresponding author: Technical University of Denmark, DTU Compute, Building 322, Room 009, DK-2800 Lyngby, Denmark, Tel: +45-45253734 huge amounts of the logged or observed data. The risk of both false positives and false negatives easily becomes too high.

What seems a viable option is to analyse an organisation's vulnerability to insider threats before an attack, or to use tool support after an attack to narrow down which actions might have occurred as part of the attack [18]. To support the analysis of organisation, several system models have been introduced that model organisations' infrastructure and actors. Examples for such models include ExASyM [18], Portunes [7], and ANKH [17]. All these models follow similar ideas, namely the modelling of infrastructure and data, and analysing the modelled organisation for possible attacks.

In all the system models mentioned above, the attacker is assumed to be almighty: he has perfect knowledge about the infrastructure, he knows the policies enforced, and he knows where data and keys are located. This can be compared to white box-testing of software components or Dolev-Yao attackers in protocol analysis [8]. Assuming so strong an attacker model implies that the system models only can give upper bounds for vulnerabilities, since the analysis has to assume that all actions that are allowed to be performed in the model also will be performed by some actor.

The system models mentioned above have another, more important shortcoming: the attacker model and the attacker behaviour are tightly integrated into the system model and the analysis. This is an undesirable property; it means that experimenting with different (insider) behaviours and types of insiders is close to impossible [4]. It is exactly this tight integration that hampers the models' applicability to analyses of an organisation's vulnerability to different kinds of attacks, to analyses of the effect of changed employee behaviour.

In this work we propose a *modular* system model that makes actor behaviour an explicit component of the model. This modularisation provides the necessary support to investigating the effect of different kinds of behaviours. By using this modular approach, system models will be able to perform new kinds of behaviour-based analyses. But the modular system model also supports implementing the analyses and behaviours that the current models provide, thus enabling a smooth transition.

The rest of this paper is structured as follows. After an overview of system models in the next section, we discuss different kinds of insiders in Section 3. In Section 4 we present the new, modular design of system models, and describe our experiences with the application of this design to a new version of ExASyM in Section 5. Section 6 concludes the discussion, and presents current and future research directions.

2 Background

In this section we briefly describe the three main formalisms for modelling organisations and analysing security threats in them: ExASyM [18], Portunes [6], and ANKH [17]. The section concludes with a discussion of the limitations of those formalisms.

The ExASyM modelling formalism was initially designed with a focus on modelling not only the logical structure and the actors interacting with or within a system, but also the *physical* infrastructure of the system, enabling convenient and precise modelling of organisations and their physical environments. ExASyM models consist of separate domains for infrastructure (physical objects), actors (processes and humans), and data items (data and keys) [18, 19]. Elements that belong to different domains can be connected: actors can be at different locations or move between them; data can be carried by actors or be placed in a location. A number of analyses can be applied to ExASyM models to reason about different security properties [18], including graph-based analyses and fully automated static program analyses [19]. The semantics of ExASyM is formalised using a variant of the Klaim family of process calculi [1, 5, 10]. In addition to facilitating formal reasoning and proofs of correctness, the formal semantics also provides a foundation for developing advanced analyses and the application of formal methods.

Portunes is another formalism for modelling systems that comprises the physical, digital, and social domains into a single environment [6]. Similar to ExASyM, the formal semantics is based on the Klaim family of languages [1, 5, 10]. Accordingly, models in Portunes consist of nodes, actions, and processes. However, it extends the Klaim base-calculus with support for mobility of nodes instead of processes. It also uses the concept of layers and containment in order to represent to which domain an actor or data element belongs. The social domain is represented by low-level policies that describe under what conditions a person trusts another person. In this way one can delegate a task to someone else; in Portunes this is used to introduce a concept for representing social engineering.

The ANKH model [17] was inspired by the actor-network theory of Bruno Latour [13, 14]. In contrast to the above two models, it has a flat structure: there is no difference between locations, objects, and data. The formal representation is a hypergraph where the hyperedges describe groups of entities that have direct access to each other, *e.g.*, all entities that are in the same room. The modelling formalism is heavily based on policies, which therefore must be well defined in order to avoid unrealistic cases. For example, it should not be possible, if having two walls in the same room, one of them to gain access to another room through the other wall.

All the aforementioned modelling languages have a common characteristic that we address in our work: the analyses they provide are not modular. This means that it is not easily possible to extend or parametrize the analyses with, *e.g.*, new behaviours, because the behaviour assumed by these models is deeply embedded into the models and analyses. Also, new model properties can be difficult to add, but these shortcomings seem less important as they in principle can be overcome by extending the language. In ExASyM, for example, it is difficult to model a physical object that changes its owner. Even though it would be possible to represent the action, there is no straightforward way to model the movement of objects since objects cannot move with actors. In Portunes and ANKH, on the other hand, there is no direct way to model dynamically produced artifacts, *e.g.*, a prank email, which makes it difficult to model certain attack scenarios.

As just mentioned, behaviour of actors is not modular, *i.e.*, it is fixed in the models. This appears to be a major problem as, for example, it is not possible to define different types of actors' behaviour so that the vulnerabilities of the system can be examined with respect to each of them. In ANKH policies are of outmost importance. Any approach of representing human behaviour would naturally depend on such policies. The possibilities of expressing attacker's behaviour would be either heaviliy restricted or require significant amount of work in order to define good enough policies. On the other hand, well defined policies also mean high level of granularity which could be more error-prone. Once we try to generate attacks from system models, having too many details may lead to a generation of attacks that are practically impossible.

The concept of human behaviour supported in the models is rather restricted and unnatural. None of the formalisms have explicit support for modelling a wide range of social aspects. In Portunes, for example, by using low-level policies, the social aspect is limited only to delegation, but other than this it would be very difficult, if possible at all, to model an attacker's behaviour. The other models have similar restrictions, hindering modellers to represent human factor in the models. However, modelling the human factor becomes increasingly important when analysing security threats [21].

The access to certain locations or data can in all models be defined by access control policies. However, there is no clear way to define the actor's decision in case of several possible actions, neither is there a way to estimate what is the probability of a certain action to happen over time. ExASyM, for example, has a pessimistic approach: all actions that can be performed in the model, will be performed by the analysis. Portunes has an explorer style, *i.e.*, exploring all possibilities thus creating a too broad world which might lead to unrealistic scenarios or potentially high level of complexity of the analysis.

In conclusion, none of the system models discussed are flexible enough when it comes to specifying and parametrizing behaviour. What is needed is exactly modularity to make human behaviour a parameter of the modelled system.

3 Kinds of Insiders

Bishop *et al.* emphasise that the distinction between *insider* and *outsider* is not a result from a binary function [2]. Instead, it is more realistic to distinguish between different kinds of attackers with respect to the level of insiderness. These levels are based on different parameters such as access, knowledge, and trust [2]. Mundie *et al.* also introduce different components in their attempt to define an ontology for insider threat [12].

The system models discussed in Section 2 consider only one single kind of attacker. It is assumed that they know everything, *i.e.*, they represent the strongest possible attacker with respect to the model. This assumption is partly justified by the fact that it is very hard to collect data to explain human behaviour. However, as discussed above, assuming a strongest possible attacker also means that the analyses on system models will deem most organisations to be vulnerable to most attacks. This happens because an attacker with legal access to large parts of the organisation, such as a CEO or a cleaning lady, also has the possibility to attack large parts of the organisation. In real life this problem is solved by, *e.g.*, trust or background checks. In the analyses of system models, we need to assume that the actor might perform the actions, thus raising an alert. Therefor we need other concepts in system models to solve this problem [20].

As mentioned earlier, the Dolev-Yao attacker is the most powerful attacker when talking about protocol analysis [3]. In the formalisms described above the insider knows how to get to any location, for example what key is needed to open a certain door, where the key is located, and how to get it. In this case, we can think metaphorically of a Dolev-Yao insider. However, such kind of insider defines the upper bound of the attacker's abilities, which is not realistic in real life. In addition, modeling such an attacker would make the system vulnerable to most kinds of attack thus making the model useless. Instead we would like to define different types of actors, *i.e.*, actors with typical kinds of behaviour.

Some studies already exist that could provide the data to define typical kinds of insiders. Magklaras *et al.* classify insider misuse as either intentional or accidental [15]. A study shows that accidental security incidents by insiders happen more often than malicious insider attacks [11]. Existing research on personality traits in relation to insider threats introduce a classification of the insiders based on motivational categories. For instance, one category in the topology is the explorer type. Driven by curiosity, they are benign and often perpetrate without realizing an attack [22].

By defining seperate groups of attackers it would be then possible to examine how vulnerable a system is towards different behaviours. One relevant evaluation parameter is, for example, the likelihood of social engineering [16].

4 Towards new models

In this section we introduce our approach to a new, modular model structure. It is to some extent inspired by the already existing formalisms presented in Section 2, but in addition it aims at providing a more straightforward and structured way of modelling human behaviour and including it in the analysis.

The biggest difference compared to the layout of the existing system models and analysis is exactly this externalization of behaviour. While in their current version, the different models all deeply integrate the behaviour into the analysis, and as argued above make it hard or impossible to change the behaviour, we extend the model with a behaviour component, which determines the actors' behaviour whenever queried by the system model.

The other extension over existing models, and also closely related to the addition of a behaviour component, is the introduction of quantitative properties for entities such as actors, actions, and locations. Supporting quantities in the model opens opportunities for new classes of questions one can ask the model. Depending on the quantities chosen, it would allow not only to ask whether a possibility of an attack exists, but also to get estimation about, for example, the cheapest attack in terms of cost, time, or risk of detection. We believe that this extension is a step closer to a more realistic approach to system analysis for vulnerabilities, as it also enables the modeller to use parametrisation of the model with behaviour of actors.

In the rest of this section we describe these extensions in more detail, and then integrate them into ExASyM in Section 5. It is important to note that these extensions are independent from the underlying model, and can be added to any of the models discussed above, as well as to other system models.

4.1 Enhancing Models with Quantities

In their current version, the models presented in Section 2 do not support quantities other than connections and numbers of edges or paths. As soon as we want to answer more specialised questions or expect more meaningful answers than that an attack is possible or how certain keys are used in the attack, we need to provide this kind of additional information manually.

In a first step we add a set of "straightforward" metrics to the models, including:

- For actions the time to perform this action, the risk of detection when performing it, and the cost

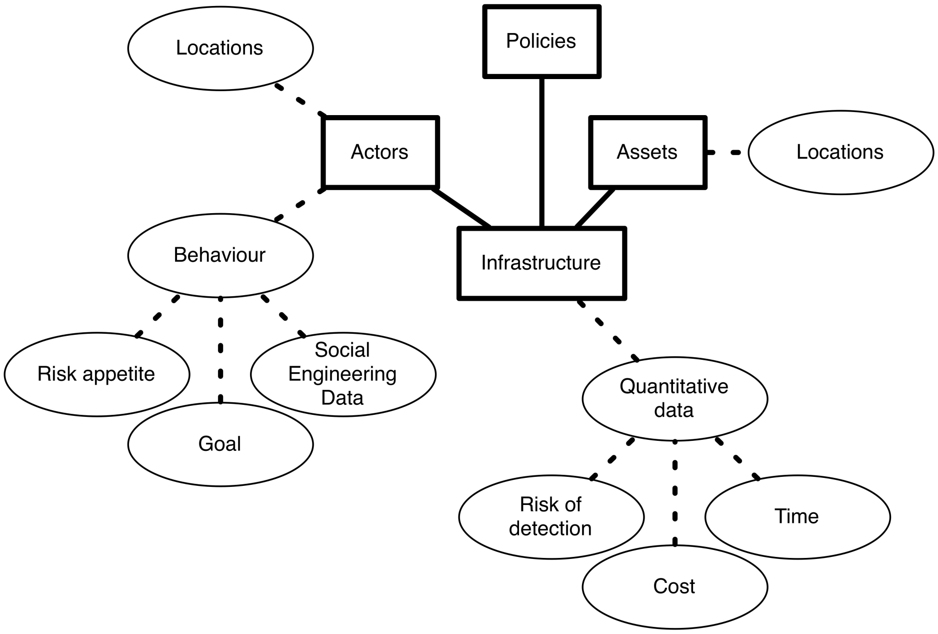

Figure 1: The new system model structure. The most noteworthy changes are the explicit behaviour component and the quantitative data, that extend the core consisting of infrastructure, assets, policies, and actors. Each actor has a separate behaviour that is queried whenever the analysis or simulation needs to perform an action. The data associated with the behaviour then determine, which action will be performed. The *goal* describes, which mode the actor is in, *e.g.*, minimizing risk of detection or time for actions. Both actors and assets are associated with locations that describe where they are located in the model.

of performing it;

- For actors the likelihood of a social engineering attack to be successful and the risk appetite of the actor; and

- For locations the risk of detection at this location (for example due to surveillance cameras).

It is important to note that most of these annotations depend on some kind of context. The time it takes to perform a certain action certainly depends on the actor, on the location, on the risk appetite, and many other factors.

We do not currently consider where these numbers come from; as mentioned before, many studies exist that analyse insider behaviour, and these can be used to provide data for the models, as could be sociological analyses and surveys. While this is ongoing work, the model in this form can already now be used for simulations and experiments.

4.2 Explicit Behaviour

The second and more fundamental change to the model structure supported in existing models is the explicit addition of a component deciding the behaviour of actors. This change can go along with a further restructuring that also introduces separate components representing infrastructure, data, and actors, respectively.

Figure 1 shows the explicit structure of the framework and the interaction between components. The behaviour component takes as input the actor, the actor's location, the data known, and the state of the system, and returns which action the actor decides to perform. Each actor has a separate behaviour that is queried whenever the analysis or simulation needs to perform an action. The *goal* describes, which mode the actor is in, *e.g.*, minimizing risk of detection or time for actions.

Models with such an explicit behaviour component ease experiments and analyses of systems with different kind of attackers, simply by replacing the behaviour. It becomes straightforward, for example, to model actors who are opportunists, who just want to obtain the item of interest and get out again, or to model behaviour driven by resource constraint such as time or risk of detection. Also the behaviour attributed to insiders, namely that they tend to repeat actions as long as they are not caught [9], and that they become less risk-aware over time, can easily be modelled.

Another interesting aspect to investigate is the effect of *mixed behaviours*, for example, an attacker minimising the risk of detection until the attack has succeeded or the attacker has been detected, and then minimising the time needed to leave the organisation.

5 Restructuring a System Model

We now apply the structure introduced in the previous section to ExASyM, to obtain a modular system model, in which the behaviour of actors is a parameter of the model. The model described here has been implemented in [23] as a restructuring of the original model [18].

After adding a behaviour component to the system, the analyses available in ExASyM had to be adapted accordingly. One of the main goals of the restructuring was to ease the addition of new analyses, and to provide APIs to support access to the model's functionality, including the behaviour. The work in [23] showed that the adaptation of the existing analyses in the new model was straightforward.

5.1 Different Kinds of Behaviours

The new behaviour component eases experiments with different kinds of behaviours, and also with phase shifts between different behaviour within the same analysis.

5.1.1 Analysing Processes

One of our first approaches to identifying insider threats in system models was the analysis of a given process describing actions of an actor in the modelled organisation [19]. This can be compared to static analysis of programs; while this approach makes sense in the case of programs, it is unclear how one would obtain the necessary process representing a complete trace of actions of an actor performed during an attack.

Representing and analysing this “predetermined” behaviour in the new, modular ExASyM are straightforward. The behaviour is instantiated with the process P that one wants to analyse, and every time the model queries for the next action of the actor, the next action in the process P is returned. In this scenario, the behaviour does completely ignore the state of the system and the location of the actor when picking the next action.

5.1.2 The Greedy Explorer

When moving away from processes, the static analysis approximates all possible actions performed by actors in the system. To do so, the analysis must assume that every action that can be performed in the model also will be performed [18]. Also this scenario is easily re-implemented using the external behaviour component. Whenever the model asks the actor for the next action, the actor gets the location where he is currently located, obtains all actions permitted at this location by the access control policy, and iteratively returns the admissible actions.

The original algorithm is shown in Figure 2 and shows the interweaving of the behaviour with the analysis. Introducing a separate behaviour component induces almost no changes in the existing analyses. At the same time it helps separating the core of the analysis (at which locations do we check for actions to perform) from the actual decision, which actions will be performed. This is an immediate result of the modularization of the system model. Another side effect is that we can simulate actors with *different* behaviours.

5.1.3 More Advanced Approaches

The real benefit of the modular behaviour component becomes apparent when considering new behaviours such as mixed behaviour with different phases, as discussed above, or history-based behaviour, which takes previous actions into account. All these could have been introduced also in the original ExASyM model, but at the cost of changing the behaviour of all actors. As part of [23] also an AI was implemented, and we are currently evaluating whether this more complex behaviour implementations contribute to more insights and a better understanding of vulnerabilities.

5.1.4 Lessons learned

The addition of an external behaviour component to ExASyM, and the reimplementation of existing analyses as well as new analyses was surprisingly easy. The code for existing analyses became much clearer, since the representations of actors and data contain all their locations, so it is the behaviour component that decides what to do in case of an actor possibly being located in different locations, not the analysis. This in itself makes adding the behaviour component a worthwhile task.

```
 1: equivalent()
 2: /* perform (log-)equivalent actions */
 3: changed = true
 4: while changed do
 5:   changed = false
 6:   for all actors $n$ do
 7:     for all locations $l$ that $n$ might be located at do
 8:       for all locations $l'$ reachable from $l$ in one step do
 9:         simulate all actions that $n$ can perform on $l'$
10:         for each action set changed if $n$ at location $l$ learns a new data item
11:       end for
12:     end for
13:   end for
14: end while
```

Figure 2: The greedy explorer algorithm presented in [18]. For each actor in the system we check for all locations he can be located at whether he can perform any actions, and if yes, perform them. When switching to the external behaviour component, the only change necessary is in line 9, where we query the behaviour for actor n at location l for actions that can be performed at l'.

```
for all actors $n$ do
  for all actions $a$ in $behaviour(n)$ do
    simulate action $a$
    for each action set changed if $n$ at location $l$ learns a new data item
  end for
end for
```

Figure 3: The reimplementation of the loop in the pseudocode in Figure 2 (lines $6-13$) using the behaviour component. It should be noted that all the information regarding, *e.g.*, possible locations of the actor is stored in its state, so the behaviour does not need additional input.

Also for developing new analyses the encapsulation of decisions about possible actions into the behaviour is beneficial, as the resulting analyses concentrate on computing the analysis result, and do not have to keep track of where actors and data might be located.

6 Conclusion

System models have recently been introduced to model organisations and evaluate their vulnerability to threats and especially insider threats. Insider threats pose difficult problems for organisations. By their very nature, insiders have access to and knowledge about the organisation's infrastructure, data, and workflows, and they use these in their everyday work to fullfil tasks. For this kind of attacks, system models are ideally suited since they can be seen as white-box testing of organisations.

In an increasing numbers of attacks also from the outside, social engineering is now used, turning an outside attack into an inside attack by convincing an insider to help; social engineering aims at making an insider perform an action that he is allowed to perform, were it part of his daily work.

To evaluate organisations with respect to both social engineering and employee behaviour in general, we need to be able to specify different kinds of behaviour for actors. However, current models do not support this explicit specification of different behaviours. Instead, behaviour is deeply embedded in the analyses supported by the models, meaning that it is a complex, if not impossible task to change behaviours. Especially when considering social engineering or the human factor in general, the ability to use different kinds of behaviours is essential.

In this work we presented an approach to making behaviour explicit. Our framework is independent of the system model used, and therefore applicable to other existing models as well. Using the external behaviour component, we not only can simulate and analyse system models with different kinds of actors, but we can also re-implement existing analyses, which without further effort take the different kinds of behaviour into account.

Acknowledgments

Part of the research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 318003 (TRE$_S$PASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

References

[1] L. Bettini, M. Loreti, and R. Pugliese. An infrastructure language for open nets. In *Proc. of the 2002 ACM symposium on Applied Computing (ACM SAC'02), Madrid, Spain*, pages 373–377. ACM, March 2002.

[2] M. Bishop, S. Engle, D. Frincke, C. Gates, F. Greitzer, S. Peisert, and S. Whalen. A risk management approach to the "insider threat". In C. W. Probst, J. Hunker, D. Gollmann, and M. Bishop, editors, *Insider Threats in Cyber Security*, volume 49 of *Advances in Information Security*, pages 115–137. Springer-Verlag, 2010.

[3] I. Cervesato. The Dolev-Yao intruder is the most powerful attacker. In *Proc. of the 16th Annual IEEE Symposium on Logic in Computer Science (LICS'01), Boston, Massachusetts, USA*, pages 16–19. IEEE, June 2001.

[4] C. Colwill. Human factors in information security: The insider threat – Who can you trust these days? *Information Security Technical Report*, 14(4):186–196, November 2009.

[5] R. de Nicola, G. L. Ferrari, and R. Pugliese. KLAIM: A kernel language for agents interaction and mobility. *IEEE Transactions on Software Engineering*, 24(5):315–330, May 1998.

[6] T. Dimkov. *Alignment of Organizational Security Policies — Theory and Practice*. University of Twente, 2012.

[7] T. Dimkov, W. Pieters, and P. H. Hartel. Portunes: representing attack scenarios spanning through the physical, digital and social domain. In *Proc. of the Joint Workshop on Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security (ARSPA-WITS'10), Paphos, Cyprus, LNCS*, volume 6186, pages 112–129. Springer-Verlag, March 2010.

[8] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, March 1983.

[9] A. L. Freitas and E. T. Higgins. Enjoying goal-directed action: The role of regulatory fit. *Psychological Science*, 13(1):1–6, January 2002.

[10] D. Gorla and R. Pugliese. Resource access and mobility control with dynamic privileges acquisition. In *Proc. of the 30th International Colloquium on Automata, Languages and Programming (ICALP'03), Eindhoven, The Netherlands, LNCS*, volume 2719, pages 119–132. Springer-Verlag, June-July 2003.

[11] I. Grant. Insiders cause most IT security breaches. `http://www.computerweekly.com/news/1280090551/Insiders-cause-most-IT-security-breaches-study-reveals`, 2009. [Online; accessed 12-August-2013].

[12] C. Huth, D. Mundie, and S. Perl. Toward an Ontology for Insider Threat Research: Varieties of Insider Threat Definitions. In *Proc. of the 3rd Workshop on Socio-Technical Aspects in Security and Trust (STAST'13), Tulane University, New Orleans, LA, USA*. IEEE, June 2013.

[13] B. Latour. On actor-network theory. *Soziale Welt*, 47(4):369—-381, December 1996.

[14] B. Latour. *Reassembling the social: an introduction to actor-network-theory*. Oxford University Press, 2005.

[15] G. Magklaras and S. Furnell. Insider threat prediction tool: Evaluating the probability of IT misuse. *Computers & Security*, 21(1):62–73, 2001.

[16] K. Mitnick and W. Simon. *The Art of Deception: Controlling the Human Element of Security*. Wiley, 2003.

[17] W. Pieters. Representing humans in system security models: An actor-network approach. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 2(1):75–92, March 2011.

[18] C. W. Probst and R. R. Hansen. An extensible analysable system model. *Information Security Technical Report*, 13(4):235–246, November 2008.

[19] C. W. Probst, R. R. Hansen, and F. Nielson. Where can an insider attack? In *Proc. of the 4th International Conference on Formal Aspects in Security and Trust (FAST'06), Hamilton, Ontario, Canada, LNCS*, volume 4691, pages 127–142. Springer-Verlag, August 2007.

[20] C. W. Probst and J. Hunker. The risk of risk analysis and its relation to the economics of insider threats. In *Proc. of the 8th Workshop on the Economics of Information Security (WEIS'09), University College London, England*, pages 279–299. Springer-US, June 2009.

[21] K. R. Sarkar. Assessing insider threats to information security using technical, behavioural and organisational measures. *Information Security Technical Report*, 15(3):112 – 133, 2010.

[22] E. D. Shaw, J. M. Post, and K. G. Ruby. Inside the mind of the insider. Security Management, December 1999.

[23] N. Thykier. A reusable framework for analysing system models. Master's thesis, DTU Compute, Technical University of Denmark, 2013.

Author Biography

Marieta Georgieva Ivanova is a PhD student in the Language-Based Technology section of the Department of Applied Mathematics and Computer Science at the Technical University of Denmark. She holds a MSc degree in Digital Media Engineering obtained at Technical University of Denmark. Her current research is focused on investigating how to model socio-technical aspects of organizational security within the TREsPASS project.

Christian W. Probst is an Associate Professor in the Department of Applied Mathematics and Computer Science at the Technical University of Denmark, where he works in the section for Language-Based Technologies. The motivation behind Christian's research is to realize systems with guaranteed properties. An important aspect of his work are questions related to safety and security properties, most notably insider threats. He is the creator of ExASyM, the extendable, analysable system model, which supports the identification of insider threats in organisations. Christian has co-organized cross-disciplinary workshops on insider threats and has co-edited a book on the topic.

René Rydhof Hansen is an associate professor at the Department of Computer Science, Aalborg University, Denmark. His research interests include security, static analysis, software verification and validation, real-time systems, and programming languages.

Florian Kammüller is Senior Lecturer in the Department of Computer Science at Middlesex University London and Privatdozent (honorary professor) at Technische Universität Berlin. He holds a PhD from the University of Cambridge and a Habilitation from Technische Universität Berlin. His reseach is centered around applications of interactive theorem proving and model checking to problems from security and software engineering. These applications range from developing security type systems for active object languages with Isabelle/HOL to verifying secure protocols for the internet, like the secured domain name system DNSsec or social networks.