Understanding How Components of Organisations Contribute to Attacks

Gu, Min; Aslanyan, Zaruhi; Probst, Christian W.

Published in:
Proceedings of the 21st Nordic Conference on Secure IT Systems (NordSec 2016)

Link to article, DOI:
10.1007/978-3-319-47560-8_4

Publication date:
2016

Document Version
Peer reviewed version

Understanding how Components of Organisations contribute to Attacks

Min Gu, Zaruhi Aslanyan, and Christian W. Probst

Technical University of Denmark
s146723@student.dtu.dk {zaas, cwpr}@dtu.dk

Abstract. Attacks on organisations today explore many different layers, including buildings infrastructure, IT infrastructure, and human factor – the physical, virtual, and social layer. Identifying possible attacks, understanding their impact, and attributing their origin and contributing factors is difficult. Recently, system models have been used for automatically identifying possible attacks on the modelled organisation. The generated attacks consider all three layers, making the contribution of building infrastructure, computer infrastructure, and humans (insiders and outsiders) explicit. However, this contribution is only visible in the attack trees as part of the performed steps; it cannot be mapped back to the model directly since the actions usually involve several elements (attacker and targeted actor or asset). Especially for large attack trees, understanding the relations between several model components quickly results in a large quantity of interrelations, which are hard to grasp. In this work we present several approaches for visualising attributes of attacks such as likelihood of success, impact, and required time or skill level. The resulting visualisations provide a link between attacks on an organisations and the contribution of parts of an organisation to the attack and its impact.

1 Introduction

Modern organisations are complex entities. Understanding the interactions between the organisation's infrastructure, IT system, and human actors is difficult; understanding possible attacks on the organisation even more so. Traditional risk assessment methods describe processes that can be used to identify attacks, and to explain the attacks' potential impact on the organisation. However, the focus of these techniques is often rather technical and ignores the internal structure and functioning of the organisation.

To improve the scope of risk assessment and the level of scrutiny, security researchers have suggested socio-technical security models, which include the physical, virtual, and social layer of organisations. Socio-technical security models acknowledge the need of considering all these levels in assessing the risk faced by an organisation since an increasing number of attacks today do involve attack steps on all three levels. The recent attack on a German steel mill [1], for example, started with a spear phishing campaign, installing malware that gave the

attackers access to the office network, and from there to the industrial control system. Eventually, the attack is said to have caused physical damage to the mill's production system.

To communicate the attacks identified in an organisation, attack trees [2, 3] are often used; due to their relatively loose definition, attack trees can be adapted to the requirements in many different settings. Attack trees provide structure to the represented attacks by relating a node representing the goal of an attack with different alternative or required sub-goals, which an attacker may or must perform. This structure makes attack trees also an appropriate target for automated identification of attacks [4–6].

The TRE$_S$PASS project [7] applies attack trees as an intermediate representation of attacks. Attacks are generated from a socio-technical system model [8,9] and are the basis of computing the risk faced by an organisation if one or more of the identified attacks are realised. Properties of interest of these attacks include required resources, such as time or money, likelihood of success, or impact of the attack. The analyses also identify the Pareto frontier of incomparable properties, for example, the likelihood of success of an attack, and the required budget.

When communicating the result of risk assessment, two components are of interest: the actual attacks and the contribution of components of the organisation under scrutiny to these attacks. While properties of attack trees or other attack models can be visualised in enlightening ways [10], the same does not hold for the connection between components of the organisation and the attack. Another limiting factor is the sheer size of attack trees, which easily can contain several thousands of nodes. Manual assessment of the individual attacks in huge attack trees is often impossible.

The generated attacks make the contribution of building infrastructure, computer infrastructure, and humans (insiders and outsiders) to the attack explicit. However, this contribution is only visible in the attack trees as part of the performed steps, for example, as leaf labels. Mapping this back to the system model is in principle not complicated. However, the actions usually involve several elements (attacker and targeted actor or asset) that may be located far apart in the model. Especially for large attack trees, visualising these relations quickly results in a large quantity of interrelations, which are hard to grasp.

In this work we present several approaches for visualising attributes of attacks such as likelihood of success, impact, and required time or skill level. The resulting visualisations provide a link between graphical attack models and graphical system models. After a discussion of visualising properties of attack trees, we present our approach of using metrics to identify the importance or contribution of parts of the attack tree, and mapping it to the system model. Our approach currently only considers contribution of model elements – it not, for example, include information on how assets and actors are used in an attack.

Our approach is independent of the attack model or socio-technical system model used. The only requirement is that all model elements have unique identifiers that establish the link between their occurrences in the attack tree and the model, respectively. While we present them in the setting of the TRE$_S$PASS

model, which is similar to ExASyM [11] and Portunes [12], the general approach can be applied to any graphical system model and any attack model. For example, the metrics used for visualising model components can also be output as a text file for sorting and further analysis.

The rest of this article is structured as follows. The next section gives an overview of graphical models for systems and attacks, followed by a description of the visualisation of properties of attack trees in Section 3. Based on these properties, we specify in Section 4 metrics for identifying the contribution of components of organisations to the attacks, and show their application in visualising the contribution to attacks. Finally, Section 5 concludes the paper and discusses future work.

2 System and Attack Models

Before discussing the contribution of components of organisations to attacks, we briefly summarise the system and attack models we consider in our work. As stated above, our approach is not limited to specific models for systems and attacks. We only require system models to provide unique identifiers for model elements, and attack models to use these identifiers in describing attack steps.

2.1 System Models

System models include representations of both the physical and the digital infrastructure of an organisation. Approaches such as ExASyM [11] and Portunes [12] represent relevant elements as nodes in a graph. Nodes represent locations, actors, processes, and items, and can be annotated with policies. Actors, processes, and data are located at locations, items and data can also be contained in another item. In our abstraction of the model, these nodes represent the organisational components that enable and contribute to attacks. All elements in the model provide a unique identifier that can be used to refer to the element and to obtain, for example, information on its concrete type, model, or other relevant properties. This information is used in the attack generation, but it can also provide input to the visualisation of system models, for example, whether two elements should be connected by an edge (*e.g.*, two locations) or one within the other (*e.g.*, two items).

While models such as ExASyM [11] and Portunes [12] also define actions that can be performed by actors and processes, these are not required for our approach. We only expect to be able to extract actors and arguments of actions from leaf nodes in attack trees.

2.2 Attack Models

Similarly, attack models represent possible attacks on the modelled organisation. For the approach in this paper, we only require that attack goals can be divided into sub-goals that can be combined either *conjunctively* (must all be completed)

or *disjunctively* (only one sub-goal need to be completed). This is very similar to attack trees [2, 3], and just as for these it would be interesting to allow more complex combinations at a later point.

As mentioned before we require the attack model to support extraction of actor and assets from the actions in an attack tree. In our current work, actions are contained in the attack-tree leafs. The leaf labels contain words from a regular language that provides, for example, information about type of action, performing actor, which asset is obtained, and where the asset is obtained from. The arguments to the action or exactly the identifiers that connect the attack tree with the system model. We do not need to impose other assumptions that are often found, *e.g.*, about the ordering of sub goals from left to right; this is due to the flow-insensitive nature of our visualisation.

2.3 Running Example

We use the same running example in this paper as in [5], which is based on a case study in the TRE$_S$PASS project [7] centred around an actor Alice, who receives some kind of service, *e.g.*, care-taking, provided by an actor Charlie.

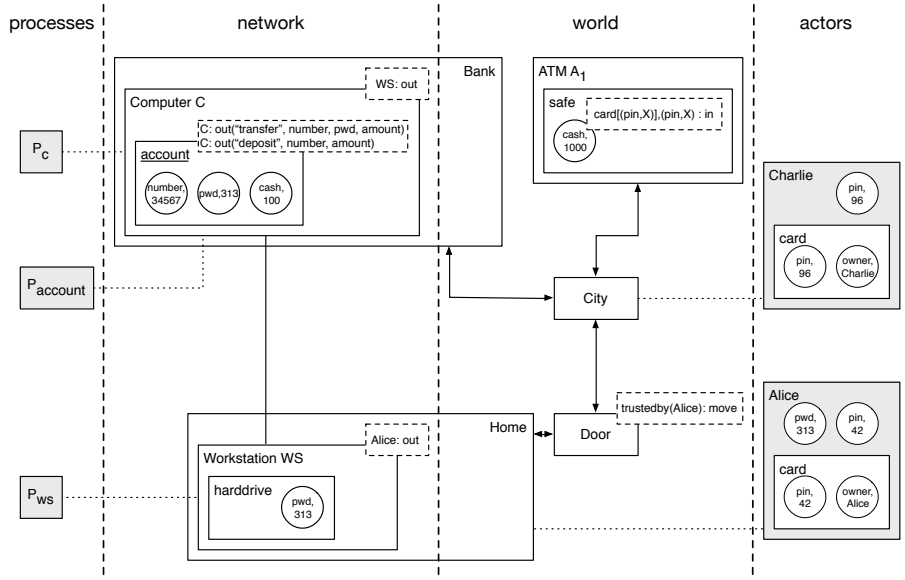

Fig. 1. Graphical representation of the example system. The white rectangles represent locations or items, the gray rectangles represent processes and actors; actors contain the items or data owned by the actor. The round nodes represent data. Solid lines represent the physical connections between locations, and dotted lines represent the present location of actors and processes. The dashed rectangles in the upper right part of some nodes represent the policies assigned to these nodes.

Charlie's employer has a company policy that forbids him to accept money from Alice or to steal money. Figure 1 shows a graphical representation of the example scenario, consisting of Alice's home, a bank with an ATM, and a bank computer. Alice owns a card and a concomitant pin code to obtain money from an ATM, and a password to initiate transfers from her workstation via the bank computer. Some of the nodes are labelled with policies in dashed boxes; for example the money at the ATM requires a card with a pin code, as well as that very pin code in order to obtain money (modelled as input).

Figure 1 shows a graphical representation of the model of our running example. The locations, represented by small rectangles, are connected through directed edges. Actors are represented as rectangles with a location, *e.g.*, Alice is at home and Charlie is in the city. Both actor nodes and location nodes can contain data and items represented as circles. In our example, Alice has a card that contains a pin code and Alice also has (knows) the pin code for her card. Actor nodes can also represent processes running on the corresponding locations. The processes at the workstation and the bank computer represent the required functionality for transferring money; they initiate transfers from Alice's home (P_{WS}), and check credentials for transfers (P_C).

Note that all elements have either a unique name or a unique value, which serve as their identifiers. If an element occurs more than once, for example, the password $(pwd, 313)$ or the Alice's pin $(pin, 42)$, these occurrences represent copies of the same artefact.

3 Visualising Attacks

The analytic risk assessment based on socio-technical security models operates on attack trees and judgments about quantitative properties of the actions performed and the actors performing them. After briefly discussing how to evaluate attack models, we present a simple approach for visualising several, potentially incomparable properties of such models. The approaches discussed in this section provide the input for the attribution of contribution of organisational components to attacks in the next section: the colouring will be used for identifying important parts of the organisation, and the analyses results provide input to the assessment of the contributions.

3.1 Evaluating Attack Models

The attack models generated from system models form the basis of analytic risk assessment. Properties of interest [13] of these attacks include required resources, such as time or money, likelihood of success, or impact of the attack based on annotations of the leaf nodes in attack trees. Analyses [14] also identify the Pareto frontier of incomparable properties, for example, the likelihood of success of an attack, and the required budget.

The mapping of actions to metrics can again be achieved by mapping the action and its arguments to a specific value. These metrics can represent any

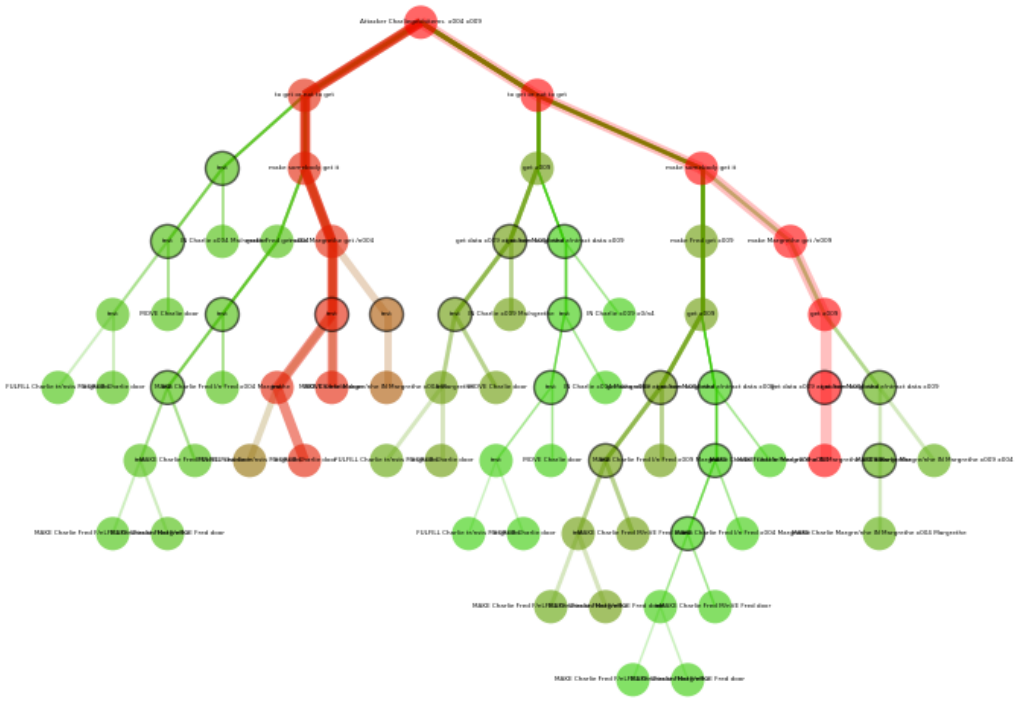

Fig. 2. Attack tree visualisation plot. Nodes with border represent conjunctive nodes, nodes without border disjunctive nodes. The two red paths represent the two attacks with the biggest likelihood of success. The left hand path, however, has a higher chance of success, which is represented by a higher saturation of the colours. The illegible labels for even so small an attack tree document the inapplicability of this concept to showing risk for organisations; attack trees tend to grow so large that they become unhandy and require different visualisation approaches.

quantitative knowledge about components, for example, likelihood, time, price, impact, or probability distributions. The latter could describe behaviour of actors or timing distributions. For the visualisation described in this article the mapping of leaf nodes to metrics and the analyses performed are irrelevant; we assume an attack tree and a mapping from its nodes to an analysis result. For the purpose of this work we have implemented simplified versions of [13, 14].

3.2 Attack Tree Visualisations

While not at the core of our work, we briefly discuss the mapping from attack tree analysis results to visualisations, since these map directly to the visualisation of the contribution of components of organisations to the risk faced by the organisation.

We have applied three visual styles to illustrate the influence of paths in the attack tree on the overall result for the tree:

– The line width of edges implies the resource usage of a specific path, that is, how resource demanding an attack path is. We assumed attackers always choose the path with lowest cost, lowest time consumption, and lowest difficulty to apply attacking. Thus the line width is inversely proportional to these three parameters – the lower the resource usage of a path, the more likely the attacker to take it (modulo other factors that come next).
– The transparency reflects the likelihood of success of a path in an attack. This attribute is directly defined in the weight measurement: the more transparent a path is, the lower its likelihood of success.
– The last and foremost property is color, which represents the overall impact of a path, normalised to percentage of the highest impact for the whole attack. The impact value is determined by the required resources, likelihood of success as well as the profit of the attack. In general the color scale chosen is between two colours, where one color represents 0%, the other 100%, and other values are combination of the two. In our example the color scale goes from green to red, which means the impact is increasing from low to high.

Clearly, more advanced visualisations provide even deeper insights into the scenarions represented by an attack tree. In the TRESPASS project we have explored many such methods [10].

3.3 Pareto-Efficient Solutions

In case of multiple parameters most analytical methods optimise one parameter at a time, *e.g.*, minimise cost or maximise probability of an attack. Such methods may lead to sub-optimal solutions when optimising conflicting parameters, *e.g.*, minimising cost while maximising probability; in this scenario it may not be possible to identify the attack that will result in the biggest gain for the attacker.

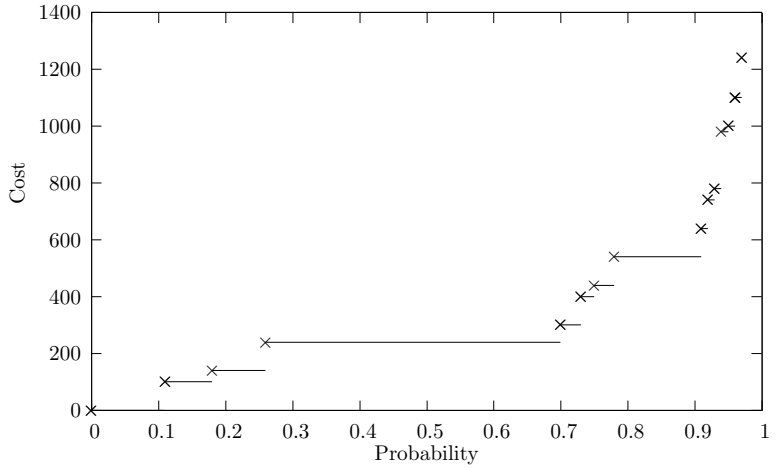

Fig. 3. Pareto efficient solutions for the attack tree [14].

Pareto-efficient solutions [14] result in combinations of these conflicting parameters, and can be used to approximate the results for comparable values. Figure 3 shows an example of a Pareto-efficient solutions for an attack tree that results in probability of a successful attacks ranging from 0 to 0.97 and the corresponding cost ranging from 0 to 695. Assuming that the attacker has a fixed budget or a rational attacker who will not launch an attack if the cost is higher than the expected gain, we can identify the optimal Pareto-efficient solution from this set.

We are currently experimenting with approaches for visualising this directly on the attack tree to indicate how close a path in the attack tree is to the most Pareto-efficient solution. The approach of scaling between two colours, which we applied for unary predicates as discussed above, does not carry over to binary predicates, where we, *e.g.*, have worse attacks (for the attacker) with higher probability. We are currently considering three colours, *e.g.*, green – red – blue, or stretching of lines, where the colours have the same meaning as discussed above, but the path lengths are scaled depending on how close they are to the Pareto-efficient solution. The scaling would make identification of the most Pareto-efficient attack (and the ordering on attacks) straightforward, since the longer paths are more efficient.

4 Contribution of Components of Organisations to Attacks

Now we put the different elements described above together to visualise the relation between attack trees and system models. Remember that we require all elements in the model to have unique identifiers; we use this identifier to associate model components and attack tree actions.

As for attack trees we need a measure for how much a model element contributes to a given attack. We apply techniques similar to our earlier work on insiderness [15].

4.1 Measuring Impact

Computing the actual impact of a model component on an attack is as difficult as computing the impact of an attack; the results can be used for ordering attacks or influence, but they should not be taken as absolute answers. With this in mind we have applied several techniques for measuring the impact of components on attacks.

As mentioned before we require the attack model to support extraction of actor and assets from the actions in an attack tree, and actions are contained in the attack-tree leafs. Leaf labels provide information about type of action, performing actor, which asset is obtained, and where the asset is obtained from. All this information is provided through the identifiers that connect the attack tree with the system model.

4.2 Counting Occurrences

The simplest concept of measuring impact is that of *counting occurrences* of identifiers. It computes for a given entity in how many places it contributes to the whole attack tree or a path. The occurrence-based ranking ignores analysis results such as impact or likelihood. It is either measured as absolute number or as percentage of occurrences of identifiers in the path or tree being analysed. It is computed per identifier id for a set of nodes in a subtree of the attack tree that represents an attack, assuming that $id \in S$ returns 1 if true, and 0 otherwise, and that node n has successors $c \in succ(n)$:

$$\mathcal{I}(id, n) := \begin{cases} [x, x] & x = (id \in actor(n)) + (id \in assets(n)), \text{ if } n \text{ is a leaf node} \\ [l, u] & l = min(\mathcal{I}(id, c)), u = max(\mathcal{I}(id, c)), \text{ if } n \text{ is a disjunctive node} \\ [l, u] & l = \Sigma_c\{l' | [l', \_] = \mathcal{I}(id, c)\}, u = \Sigma_c\{u' | [\_, u'] = \mathcal{I}(id, c)\}, \text{ if } n \text{ is a conjunctive node} \end{cases} \quad (1)$$

As a first crude measure, this impact provides a defender with a quick overview of which components of the organisation actually occur in the attack tree.

The occurrence-based impact provides for every identifier a lower and an upper bound of occurrences; for conjunctive nodes these will be the same, for disjunctive nodes the lower bound is the minimum of the lower bounds, and the upper bound is the maximum of the upper bounds of the child nodes. The combination of lower and upper bounds provides a measure for how reliable the numbers are. It also allows to identify, whether certain elements occur in all attacks: if $\mathcal{I}(id, r) = [x, \_]$ for some identifier id, the root of the attack tree, and $x > 0$, then the element with id contains in every attack in the tree.

4.3 Weighted Sum

The impact factor based on occurrences in the generated attacks is a rather crude approximation, since every occurrence of an identifiers is assigned the same impact independent on the *actual* contribution to the attack. Given that the analyses of attack trees described in Section 3.1 provide us with quantitative information about attacks, we can improve over the occurrence-based ranking by weighting occurrences of identifiers with the impact of the attack they occur in. The factors we can choose from are limited by available analyses only, but include, for example, the likelihood of success, required time, difficulty, and cost.

In contrast to the occurrence-based impact we now *include* one of the analysis results, by weighting the count for an identifier with the weight of the path, and potentially normalising it. As before, it is either measured as absolute number or as percentage of occurrences of identifiers in a subtree of the tree being analysed.

For defining the impact, we assume for identifier id for a node n on a path in the attack tree:

- the set-membership test $id \in S$ returns 1 if id is in S, and 0 otherwise,

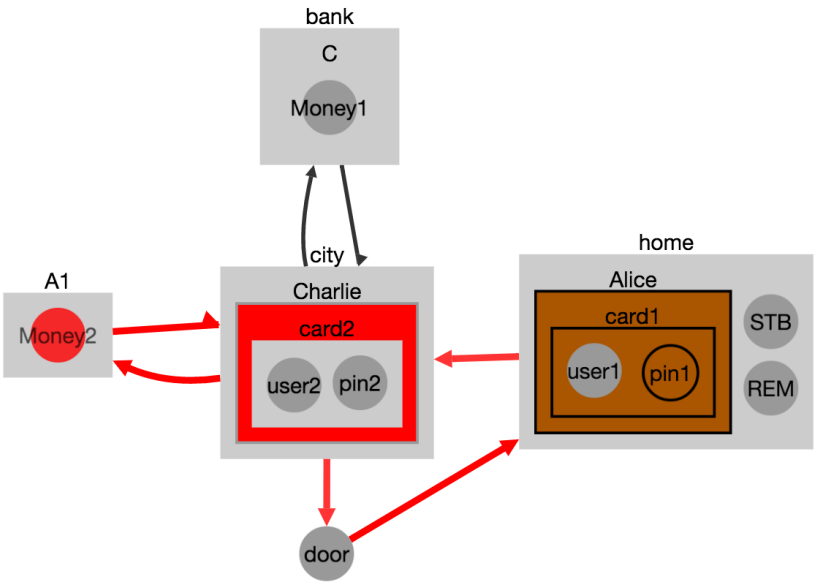

Fig. 4. Visualisation of the weighted impact of an attack tree on the physical infrastructure part of the example model from Figure 1. Charlie is identified as the major culprit as he occurs in every single attack step. Alice is less involved, since Charlie in some attacks might steal money from the ATM directly (or the ATM altogether). It is also clear from the visualisation that the user tag on Alice's card is not used in the attack, and neither is, *e.g.*, the bank computer C.

- $succ(n)$ returns the successors of node n in the attack tree, and
- $val(n, p)$ returns the result of the attack tree analysis for a node n in the (sub-)tree p.

Using these functions, we compute the contribution of an asset or actor with identifier id, at node n with subtree p rooted at n based on the following cases. If n is a leaf node, we obtain the result of the attack tree analysis for n and p. If n is a disjunctive node, we compute the minimal impact for successors of n. If n is a conjunctive node, the computation depends on the analysis result val we are using. If we measure difficulty, time, or cost, we value the impact to be the sum of the impact of all successors. If we measure likelihood of success, we assume the impact to be the minimal impact:

$$\mathcal{I}(id, n, p) := \begin{cases} v_l = val(n,p) \cdot (id \in actor(n) + id \in assets(n)) \\ \quad \text{if } n \text{ is a leaf node} \\ v_d = min_{c \in succ(n)}(\mathcal{I}(id, c, p)) \\ \quad \text{if } n \text{ is a disjunctive node} \\ v_{ca} = \Sigma_{c \in succ(n)} \mathcal{I}(id, c, p) \\ \quad \text{if } n \text{ is a conjunctive node and we} \\ \quad \text{measure difficulty, time, or cost} \\ v_{cm} = min_{c \in succ(n)}(\mathcal{I}(id, c, p)) \\ \quad \text{if } n \text{ is a conjunctive node and we} \\ \quad \text{measure likelihood of success} \end{cases} \tag{2}$$

Figure 4 shows the visualisation based on the weighted impact. For example, the impact of Alice and her card on the attacks is different from the impact of Charlie; for occurrence-based visualisations we would have expected a similar result since they do not occur in all attacks. The reason is another, though: the difference in impact is due to the fact that Charlie might decide to steal the ATM, which has lower cost and higher chance of success than, *e.g.*, social engineering Alice.

4.4 Visualising Paths

Depending on the kind of attack trees, they contain information about moves of the attacker in the organisation or not. If the move information is contained in the attack tree, then the methods above extend to visualising in the system model, which locations of the modelled organisation are most important for the attack. This information is especially interesting for deciding, *e.g.*, about the need for (better) surveillance.

4.5 Visualising Pareto-efficient Solutions

As mentioned in Section 3.3, visualising Pareto-efficient solutions requires special approaches due to the fact that the best solution may be in the middle of the spectrum of possible attacks. Therefore, it may be important to visualise not only the best solution, but also identifying solutions that are worse or better, but are not chosen due to the efficiency criterion. The three-colour option discussed above is also applicable in the model setting; other approaches such as the scaling of edges do not carry over since different attacks with differing quantitative valuations must be visualised on the same model.

4.6 Visualising different components

There exist many different analyses on attack trees, and it may be interesting to investigate and visualise several values combined on a system model. For many interesting counting approaches, *e.g.*, the ones discussed here, one can

combine different values into a vector, and apply for each value the targeted counting operation. Since the values generally may not be comparable directly, one then can either apply Pareto-based techniques, visualise the different values simultaneously, or apply a summation function that combines the individual values.

5 Conclusion

Modern organisations are complex socio-technical systems. Understanding the interactions between the organisation's infrastructure, IT system, and human actors is difficult; understanding possible attacks on the organisation even more so. While attack trees are a natural approach to communicate risks and possible attacks, it is often hard to estimate, which parts of an organisation contribute to these attacks. Even worse, attack trees tend to be so huge that they are hard to understand. Visualising the attack trees directly eases the treatment, but still leaves defenders with large trees; what is needed is an approach that relates the model to attacks.

In this article we have presented a systematic approach for such a mapping of the results of attack generation back to system models. The visualisation can be based on arbitrary counting of occurrences of model elements in the generated attacks. We have presented two such approaches based on simple occurrence and weighted occurrence. More complex ones could, *e.g.*, also take the role of actors into account, such as attacker, victim, or social engineered.

In the TRE$_S$PASS project [7], visualisations such as our approach contribute to the attack navigator [16, 17]. Beyond this, the techniques presented here are widely applicable. Our approach is agnostic to the underlying system and attack models. The only requirement is the ability to associate actions and the involved artefacts in the attack model with elements in the system model, and to obtain quantitative judgments about the attacks. While the techniques discussed in this work especially target IT security attacks, the techniques are applicable to any kind of attacks and risks.

We are currently working on further refinement of the visualisations, *e.g.*, for Pareto-efficient solutions, and on more advanced counting functions. As mentioned above, it would be interesting to take the role of an actor into account. We are also investigating how to extend our approach to attack-defence trees [18], which combine actions by attackers with mitigating actions by defenders.

Acknowledgment

Min Gu is an Erasmus Mundus student and receives funding from NordSecMob – Master's Programme in Security and Mobile Computing. 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. BBC News: Hack attack causes 'massive damage' at steel works. Available from http://www.bbc.com/news/technology-30575104. (2014) Last visited October 15, 2015.
2. Schneier, B.: Attack Trees: Modeling Security Threats. Dr. Dobb's Journal of Software Tools **24**(12) (1999) 21–29
3. Kordy, B., Pièvre-Cambacédès, L., Schweitzer, P.: Dag-based attack and defense modeling: Don't miss the forest for the attack trees. Computer Science Review **13-14** (2014) 1 – 38
4. Vigo, R., Nielson, F., Nielson, H.R.: Automated generation of attack trees. In: Proceedings of the 27th Computer Security Foundations Symposium (CSF), IEEE (2014) 337–350
5. Ivanova, M.G., Probst, C.W., Hansen, R.R., Kammüller, F.: Transforming graphical system models to graphical attack models. In Mauw, S., Kordy, B., Jajodia, S., eds.: Graphical Models for Security. Volume 9390 of Lecture Notes in Computer Science., Springer (2015) 82–96
6. Ivanova, M.G., Probst, C.W., Hansen, R.R., Kammüller, F.: Attack tree generation by policy invalidation. In Akram, R.N., Jajodia, S., eds.: Information Security Theory and Practice. Volume 9311 of Lecture Notes in Computer Science., Springer (2015) 249–259
7. The TREsPASS Consortium: Project web page. Available at http://www.trespass-project.eu. Last visited April 2016.
8. Kammüller, F., Probst, C.W.: Invalidating policies using structural information. In: 2nd International IEEE Workshop on Research on Insider Threats (WRIT'13), IEEE (2013) Co-located with IEEE CS Security and Privacy 2013.
9. Kammüller, F., Probst, C.W.: Combining generated data models with formal invalidation for insider threat analysis. In: 3rd International IEEE Workshop on Research on Insider Threats (WRIT'14), IEEE (2014) Co-located with IEEE CS Security and Privacy 2014.
10. Li, E., Barendse, J., Brodbeck, F., Tanner, A.: From A to Z: Developing a Visual Vocabulary for Information Security Threat Visualisation. In: Graphical Models for Security. (2016)
11. Probst, C.W., Hansen, R.R.: An extensible analysable system model. Information Security Technical Report **13**(4) (November 2008) 235–246
12. Dimkov, T., Pieters, W., Hartel, P.H.: Portunes: Representing attack scenarios spanning through the physical, digital and social domain. In Armando, A., Lowe, G., eds.: Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security. Volume 6186 of Lecture Notes in Computer Science., Springer (2010) 112–129
13. Lenin, A., Willemson, J., Sari, D.P.: Attacker profiling in quantitative security assessment based on attack trees. In Bernsmed, K., Fischer-Hübner, S., eds.: Nordic Conference on Secure IT Systems. Volume 8788 of Lecture Notes in Computer Science. (2014) 199–212
14. Aslanyan, Z., Nielson, F.: Pareto efficient solutions of attack-defence trees. In Focardi, R., Myers, A.C., eds.: Principles of Security and Trust. Volume 9036 of Lecture Notes in Computer Science., Springer (2015) 95–114
15. Probst, C.W., Hansen, R.R.: Reachability-based Impact as a Measure for Insiderness. In: 5th International Workshop on Managing Insider Security Threats (MIST 2013). (2013)

16. Probst, C.W., Willemson, J., Pieters, W.: The attack navigator. In Mauw, S., Kordy, B., Jajodia, S., eds.: Graphical Models for Security - Second International Workshop, GraMSec 2015, Verona, Italy, July 13, 2015, Revised Selected Papers. Volume 9390., Springer (2016) 1–17
17. Pieters, W., Barendse, J., Ford, M., Heath, C.P.R., Probst, C.W., Verbij, R.: The navigation metaphor in security economics. IEEE Security & Privacy **14**(3) (2016) 14–21
18. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack–Defense Trees. Journal of Logic and Computation (2012)