Formalizing Agents' Beliefs for Cyber-Security Defense Strategy Planning

Karsten Martiny[1], Alexander Motzek[2], and Ralf Möller[2]

[1] Hamburg University of Technology, Germany,
karsten.martiny@tuhh.de
[2] University of Lübeck, Germany,
{motzek,moeller}@ifis.uni-luebeck.de

Abstract. Critical information infrastructures have been exposed to an increasing number of cyber attacks in recent years. Current protection approaches consider the reaction to a threat from an operational perspective, but leave out human aspects of an attacker. The problem is, no matter how good a defense planning from an operational perspective is, it must be considered that any action taken might influence an attacker's belief in reaching a goal. For solving this problem this paper introduces a formal model of belief states for defender and intruder agents in a cyber-security setting. We do not only consider an attacker as a deterministic threat, but consider her as a human being and provide a formal method for reasoning about her beliefs given our reactions to her actions, providing more powerful means to assess the merits of countermeasures when planning cyber-security defense strategies.

Keywords: Adaptive Defense of Network Infrastructure, Semantic Information Representation, Situational Awareness, Epistemic Logic

1 Introduction

Critical information infrastructures have been exposed to an increasing number of cyber attacks in recent years. Cyber-physical systems in areas such as power plants or medical applications require special attention to defend them against any potential cyber attacks. It is important to note that, due to external constraints, established security measures—such as patching known vulnerabilities—are only applicable to a limited extent. Legal requirements might allow only the use of certified software versions, or new patches cannot be applied at a certain point in time because compatibility tests are still pending. A common effect is that external constraints leave critical information infrastructures exposed to known vulnerabilities, at least for some time.

An established approach to analyze threats from identified vulnerabilities is the use of attack graphs. Standard reactions to such an analysis include the proactive removal of identified vulnerabilities. However, for the intended application area of our work, these measures are usually not feasible without impairing the mission of the organization responsible for the critical infrastructure. Therefore, it is of utmost importance to carefully analyze potential consequences of applicable countermeasures or sequences of countermeasures (as part of defense strategies). Maintaining a model of an intruder's belief state provides the defender with improved means to assess the merits of potential defense strategies and novelly allows to analyze effects of taken actions on a human level.

To formalize the analysis of beliefs, we use *Probabilistic Doxastic Temporal (PDT) Logic* to represent the belief states of both the intruder and the defender. We assume that analysis starts at the last point in an attack graph where immediate consequences are pending, as described in [10,5]. It is reasonable to react as late as possible in such scenarios, as intruders (or attackers) already spent significant efforts reaching into the network. As any action—including false alarms—might impact the network in the same way as a real attack, reactive measures have to be used very carefully. The approach presented in this paper aids the defender in selecting the best countermeasure by providing means to reason about the belief states of an attacker.

The remainder of this paper is structured as follows. An overview of related work is given in Section 2. After a summary of PDT Logic in Section 3, we show how the attacker's and defender's beliefs can be formalized in Section 4 and discuss a small example to show how beliefs evolve differently, depending on the respective observations. Finally, the paper concludes with Section 5.

2 Related Work

Automatic attack graph generation has been an active topic of research. Starting in about 1998 with [11], newer contributions such as MulVal [9] and, e.g., studies such as [4] pave the way for better cyber security. Formal methods of analyzing attack graphs are introduced in [5], which will serve as a base for our example.

Attack graphs provide an excellent base for creating corresponding defense strategies, which range from analysis of efficient placement of intrusion detection systems [8] over employing an integer optimization problem in selecting the current best countermeasure [13,14] up to addressing situational awareness in quantitative scores such as [15] and validation of overall network defense as in [6]. However, indirect consequences—such as accidentally revealing information to an opponent—are not considered. We provide a formal method for analysis of indirect consequences and an automated consideration in defense planning.

Forms of intruder-defender-interactions have been studied on a personal level for instance in [12] and [16], but do only address the issue from a "psychological" point of view and do not provide means to formalize such behaviors. Studying intruders on a personal level comes along with developing attacker profiles and the distinction of different priorities and behaviors of attackers. Such profiles have notably been studied by Chiesa in [2,3], but do not provide an assessment of consequences, or even formalisms of such profiles, on an (automated) process of defense planning. [1] proposes an automated approach considering behavioral models in cyber security, but focuses mostly on intruder detection, while a suitable countermeasure selection is not achieved.

While all those fundamental pieces for an assessment of agent beliefs in defense planning exist in the literature, their interaction is not covered. This paper proposes theoretic fundamentals for a formalized assessment of agent beliefs in cyber-security defense planning.

3 PDT Logic

In order to formally represent the belief states of both the intruder and defender, we use PDT Logic [7], a formalism to represent and reason about probabilistic beliefs and their temporal evolution in multi-agent systems. This section provides a summary of the key concepts of PDT Logic that are used utilized in this work.

Syntax We assume the existence of a first order logic language with finite sets of constant symbols $\mathcal{L}_{cons}$ and predicate symbols $\mathcal{L}_{pred}$, and an infinite set of variable symbols $\mathcal{L}_{var}$. Every predicate symbol $p \in \mathcal{L}_{pred}$ has an *arity*. Any member of the set $\mathcal{L}_{cons} \cup \mathcal{L}_{pred}$ is called a *term*. A term is called a *ground term* if it is a member of $\mathcal{L}_{cons}$. If $t_1, \cdots, t_k$ are (ground) terms, and p is a predicate symbol in $\mathcal{L}_{pred}$ with arity n, then $p(t_1, \cdots, t_k)$ with $k \in \{0, \cdots, n\}$ is a (ground) atom. If a is a (ground) atom, then a and $\neg a$ are (ground) *literals*. The former is called a *positive literal*, the latter is called a *negative literal*. The set of all ground literals is denoted by $\mathcal{L}_{lit}$. *Formulas* are built using $\wedge, \vee, \neg$ as usual. $\mathcal{B}$ denotes the Herbrand Base of $\mathcal{L}$, i.e., the set of all ground atoms that can be formed through from $\mathcal{L}_{pred}$ and $\mathcal{L}_{cons}$. Time is modeled in discrete steps. Generally, the set of agents $\mathcal{A}$ may be arbitrarily large, but for this work, we assume that the set of agents consists of a intruder I and a defender D.

Observation atoms To express that some group of agents $\mathcal{G} \subseteq \mathcal{A}$ observes some fact $F \in \mathcal{L}_{lit}$, we use the notion $Obs_{\mathcal{G}}(F)$. Note that F may be a negative literal and therefore we can explicitly specify observations of certain facts being false (such as “it is not raining”). We assume that the agents in $\mathcal{G}$ not only observe that l holds, but that each agent in $\mathcal{G}$ is also aware that all other agents in $\mathcal{G}$ make the same observation. The set of all observation atoms is denoted by $\mathcal{L}_{obs}$.

Possible Worlds The concept of possible worlds describes what combinations of events can actually occur in the modeled scenario. I.e., a world consists of a set of ground atoms and a set of observation atoms, describing what events actually hold and what is observed in this world, respectively. The set of all possible worlds is denoted by $W \subset 2^{\mathcal{B}} \times 2^{\mathcal{L}_{obs}}$. If an agent is not able to differentiate between different possible worlds, we say that these worlds are *indistinguishable* to this agent. Namely, an agent i cannot distinguish two possible worlds w_1 and w_2, if both worlds contain exactly the same set of observations for agent i. We use $\mathcal{K}_i(w)$ to denote the set of worlds that agent i cannot distinguish from world w. Naturally, if i considers w as actually being possible (because it complies with all of i's observations), it also considers all worlds $\mathcal{K}_i(w)$ possible.

Threads To describe the temporal evolution of the modeled scenario, we use the concept of threads: A thread is a mapping $Th : \tau \rightarrow W$. Thus, a thread is a sequence of worlds and $Th(t)$ identifies the actual world at time t according to thread Th. The set of all possible threads is denoted by $\mathcal{T}$.

Subjective posterior probabilistic temporal interpretations Every possible thread in the modeled scenario can be associated with a probability value that describes how likely it is that the model evolves exactly according to the respective thread.

Such a probability distribution across all possible threads is called a *probabilistic interpretation*. Initially, a probability distribution over the set of threads is given by the *prior probability assessment* $\mathcal{I}$, which is the same for all agents. With the occurrence of certain observations, agents will update their respective probability assessments over the set of threads. For instance, if some agent observes a specific fact, it will only consider threads possible, which actually contain this observation, i.e., the agent's probability assessments for all other threads will be updated to 0, while another agent, who did not make this observation, might still consider these threads possible. Thus, with the evolution of time, every agent maintains *subjective* interpretations.Since these interpretations depend on the occurrence of events in a specific thread, at a single time point different interpretations could be possible, depending on the actual thread. The subjective posterior probabilistic interpretation that an agent i associates to a thread Th at time t, given that the point-of-view thread is Th' is denoted by $\mathcal{I}_{it}^{Th'}(Th)$.

Subjective posterior probabilistic temporal interpretation In the beginning the probability distribution for the threads is given by the *prior probability assessment* $\mathcal{I}$. It is the same for all agents. With the observation of an event by one or a group of agents, the interpretation for every agent needs to be updated. With agent i, time point t, and *point of view thread* Th the update rule is

$$\mathcal{I}_{it}^{Th'}(Th) = \begin{cases} \dfrac{1}{\alpha_{it}^{Th'}} \cdot \mathcal{I}_{it-1}^{Th'}(Th) & \text{if } Th(t) \in \mathcal{K}_i(Th'(t)) \\ 0 & \text{if } Th(t) \notin \mathcal{K}_i(Th'(t)) \end{cases} \tag{1}$$

with $$\alpha_{it}^{Th'} = \sum \mathcal{I}_{it-1}^{Th'}(Th) : Th(t) \in \mathcal{K}_i(Th'(t)).$$

The threads that were possible at time $t-1$ are examined, if they are still possible at time t. $\alpha_{it}^{Th'}$ is the sum of the probabilities at time $t-1$ of all possible threads at time t. These probabilities are divided by $\alpha_{it}^{Th'}$ and this leads to the new probability distribution, the *subjective posterior probabilistic temporal interpretation* $\mathcal{I}_{it}^{Th'}(Th)$ at time t of agent i. We assume a synchronous system, so the agents can distinguish between the worlds $Th(t)$ and $Th(t-1)$ even if they made no observation.

Belief in ground formulae $B_{it'}^{lu}(F_t)$ is a *belief formula* indicating that an agent i believes with a probability in a range of $[l, u]$ that a formula F, which was satisfiable at time t, still holds at time t'

$$\mathcal{I}_{it'}^{Th'} \models B_{it'}^{lu}(F_t) \quad \text{iff} \quad l \leq \sum_{Th \in \mathcal{T}, Th(t) \models F} \mathcal{I}_{it'}^{Th'}(Th) \leq u. \tag{2}$$

Nested beliefs A nested belief is the belief of an agent in another agent's belief. Agent i believes at time t' with a probability in the range $[l, u]$ that agent j believes at time t in a belief formula B with a probability in the range of $[l_j, u_j]$

$$\mathcal{I}_{it'}^{Th'} \models B_{it'}^{lu}(B_{jt}^{l_j u_j}(F)) \quad \text{iff} \quad l \leq \sum_{\substack{Th \in \mathcal{T} \\ \mathcal{I}_{jt}^{Th} \models B_{jt}^{l_j u_j}(F)}} \mathcal{I}_{it'}^{Th'}(Th) \leq u. \tag{3}$$

4 Formalizing Agents' Beliefs

4.1 Considerations on the Target Domain

As discussed in Section 1, we are concerned with situations where preventive security measures are not always an option. Thus, the network might be exposed to known vulnerabilities and the defender is left with choosing the best reactive countermeasure in case of an attack. Since we start our analysis at the last point in an attack graph, we have to assume that any attacker breaching this point is highly skilled (e.g., as described in [2]) and has already obtained extensive information about our network.

Any attack to the network consists of (at least) two actions: First, the attacker has to gain access to a target system with appropriate privileges. Then, custom code can be executed on this system to reach the attacker's actual goal. We do model the details of these steps, but abstractly represent the first step as an *attack* on a system resulting in a gained *shell* (e.g., through exploitation of known vulnerabilities), and the second step as some *code execution* on the target system. After having successfully obtained a shell on the target system, the attacker basically has two options: either she can proceed with the second stage of her attack (i.e., code execution) or she can try to gain access to further systems. Both options come with advantages and drawbacks for the intruder: continuing to attack further systems might result in additional compromised systems, but at the same time decreases the chance of performing an attack undetected. The choice of action depends on the attacker's actual goal; she might even attack another system without actually executing code there, but only to create distractions from her actual goal.

A network based intrusion detection system (IDS) can be used to detect attack actions on specific systems. However, in practice no IDS is perfect, i.e., both false alarms and missed attacks have to be considered when employing an IDS. This is an important point when planning defense strategies: if every detected attack is countered with a corresponding defense action, the lack of such a defense lets the attacker *know* that her attack went undetected and she might proceed with executing malicious code without having to fear any actions from the defender. Furthermore, deliberately letting the attacker execute code on a non-critical target host can provide valuable insights: an analysis of the executed code will reveal the actual goal of the attack and might further reveal the identity of the attacker. Another reason for refraining from a defense operation is that this action (e.g., unplugging a control server) might impact the mission success just as much as an attack. Consequently, deliberately letting an observed attack pass undefended might provide higher expected utility for the defender. By analyzing the potential evolution of the attacker's belief states, the choice of not defending can even be used to drive the attacker to false conclusions regarding her success.

Continuing these considerations, it might prove useful for the defender to maintain some kind of "honeypot" within the network. In its classical form, a honeypot is a system that has no productive meaning but is used instead to attract attackers and thereby provides means to analyze their goals and identities. However, since in our scenario we are dealing with highly skilled attackers, we have to assume that they would be able to identify such a honeypot immediately. Still, we can adapt the concept of honeypots to our model by maintaining backup devices of critical systems. These backup devices are disconnected from the physical world but otherwise indistinguishable from the actual productive system. This way, an intruder does not know which one the critical system is, but if she executes malicious code on the honeypot, she is not able to impact the mission success, but instead unknowingly provides the defender with the possibility to analyze the code and identify the attacker.

4.2 An Exemplary Domain Model

In the following, we introduce a small example to show how we can formally model potential attacks in a computer network and apply PDT Logic to analyze the evolution of the agents' belief states.

As explained in Section 3, our scenario contains two agents, the defender D and the intruder I. Following the considerations from the previous section, we assume that two systems are present in this network: some honeypot A and the corresponding critical system B. Possible actions on a system X are denoted by *attack(X)* and *defend(X)* with the obvious meanings. Furthermore, execution of malicious code on a system X is denoted by *exec(X)*. Finally, we have observation atoms such as $Obs_D(attack(A))$, indicating that the defender observed an attack on system A.

Building on these events, we can construct a set of threads representing all possible event sequences in this example. The resulting set is depicted in Figure 1.

This model represents our considerations from the previous section: analysis starts at some time when no attack has occurred yet ($t = 0$). Possible subsequent events are then attacks on system A or system B) (represented through nodes 53 and 78 in the graph) or no attacks (node 1). If an attack has occurred on, say, system A, the IDS can detect this attack (i.e., an $Obs_D(attack(A))$ occurs, represented through the solid outgoing edges from node 53), or the attack is not detected (represented through the dashed line). For an undetected attack, the defender obviously has no options to defend against this. For an observed attack, the defender can choose between defending against this attack (node 55) or deliberately refrain from a defense (node 56). A defense forces the intruder to abort his attack (with potential downsides to the defender's mission). Lack of a defense action gives the intruder two options again: she can execute her malicious code on the attacked system (nodes 66 and 77), or she can proceed to attack the other system (nodes 57 and 68). After a second attack, possible subsequent events match the ones discussed for the first attack. Finally, if the first attack has not been defended, there are various options for the intruder to execute malicious code: if the second attack is defended, the attacker can execute the code only on the previously attacked system (nodes 63 and and 74), otherwise she can choose between executing code only on the previously attack system (e.g., node 61) or on both systems (e.g., node 62). If an attack has been detected by the IDS, the defender is able to observe these code executions (denoted in black), for undetected attacks, subsequent code executions will remain

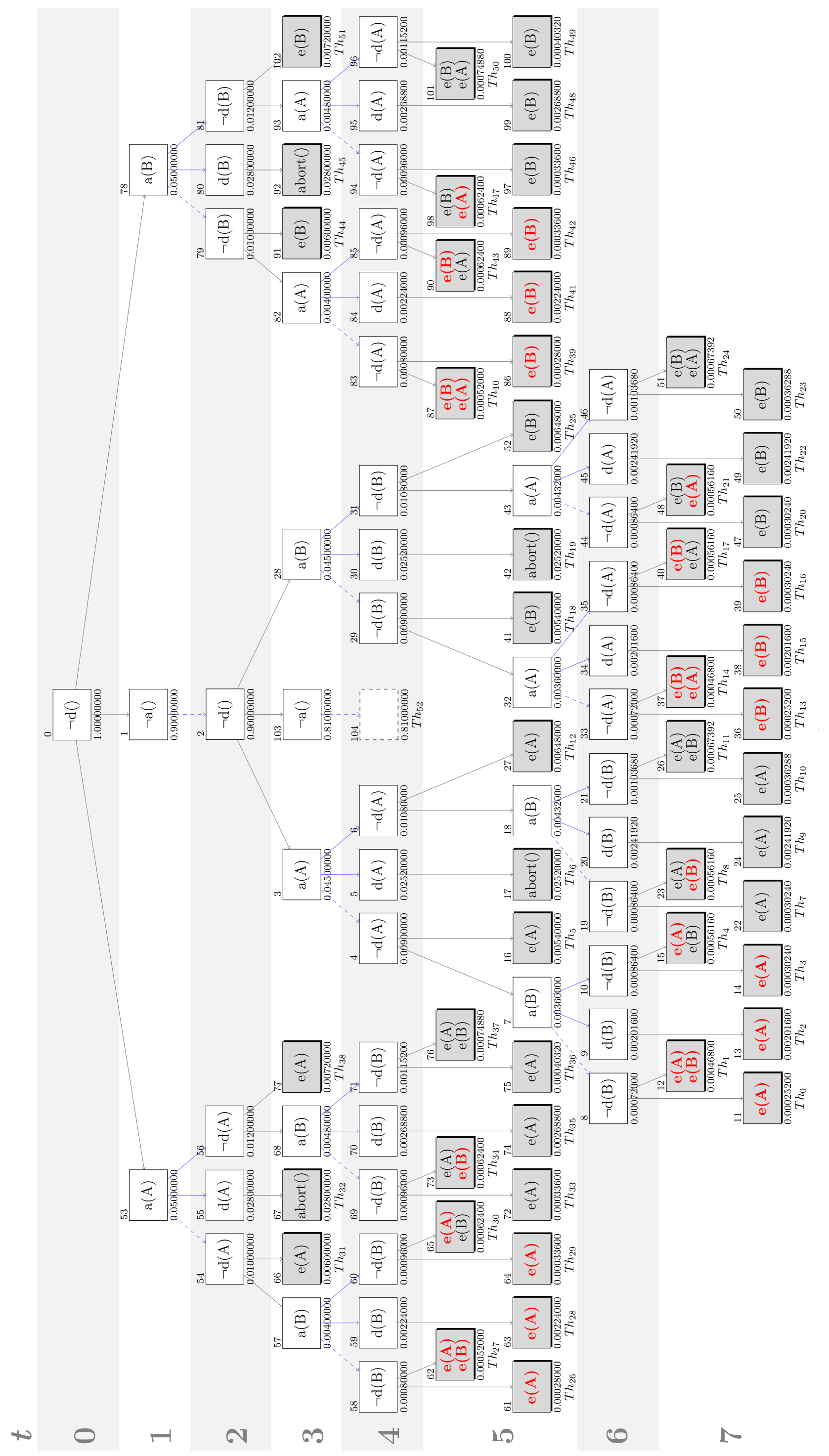

Fig. 1. Possible threads for the example domain. a(X), d(X), and e(X) denote attack, defend and code execution actions on node X, respectively. A defender's observation of an attack a(X) is marked through a solid edge (blue), a lack of an observation with a dashed edge. Unobserved code executions are marked in bold-red **e(X)**, otherwise they are marked roman-black e(X). Terminal nodes of threads are marked in gray.

undetected as well (marked in bold red in the graph). If no attacks occur at all, the system continues in its normal state as indicated through node 104—allowing for equivalent branchings of the graph if attacks occur at later points in time.

Note that in most states of this model, none of the agents has complete knowledge of the world. For instance, if the defender does not observe an attack at time $t = 2$, he cannot distinguish between undetected attacks on either system or the actual absence of an attack (i.e., he considers nodes 54, 2, and 79 as actually being possible). If the intruder actually attacked a system and does not observe a defense action, she does not know whether her attack was actually undetected or whether it was observed and intentionally not defend (i.e., she is unable to distinguish between nodes 54 and 56, or 79 and 81, respectively).

To assign probabilities to every thread in this scenario, we start with assigning probabilities to single events. Then, we can determine the probability of a thread as a joint probabilities of the respective events contained in this thread. Reasonable values of single events corresponding to the considerations in Section 4.1 are given in the following table. The resulting probabilities of individual nodes are depicted in Figure 1.

Table 1. Single event probabilities for the example

Event	Probability
Attack on a specific system at a specific time point	0.05
Attack detected by the IDS	0.80
Defend action after an observed attack	0.70
Second attack after a successful first attack	0.40
Code execution on both nodes after two attacks	0.65

4.3 Evolution of Beliefs

To illustrate how we can use this example to analyze the evolution of both agents' beliefs, let us assume that the intruder has attacked system A at time $t = 3$ (i.e., the actual world at $t = 3$ is represented through node 3). If the defender has observed this attack, he has to decide whether to defend the system against this attack or not. However, even if the defender observed this attack, he is not able to identify the actual node in the attack graph, because he is unable to distinguish between the situation where $attack(A)$ is the first action (node 3) and the situation where this was preceded by an undetected attack on B (node 82). To analyze the defender's expectations for the worst case (an unobserved code execution on the productive system B, i.e., $exec(B) \wedge \neg Obs_D(exec(B))$, one can analyze the threads containing nodes 5 or 84, and 6 or 85 in Figure 1, respectively. By summing over the normalized probabilities of these respective threads, where the terminal node contains $exec(B) \wedge \neg Obs_D(exec(B))$, one can verify that the following holds:

$$\neg defend(A) \text{ at } t = 4 \quad \models \; B_{D,4}^{0,\,0.05}(exec(B) \wedge \neg Obs_D(exec(B))), \text{ and}$$
$$defend(A) \text{ at } t = 4 \quad \not\models \; B_{D,4}^{0,\,0.05}(exec(B) \wedge \neg Obs_D(exec(B))).$$

I.e., the defender has lower expectations in the worst case actually happening if he choses to defend this attack. Consequently, we assume that the defender decides against a defend action, and we consider node 6 as the actual node for the following discussions. Next to protecting the system against the worst case, the defender is also interested in getting opportunities to analyze the intruder's malicious code. We can express his beliefs in observing a code execution (i.e., $Obs_D(exec(A)) \vee Obs_D(exec(B))$ as

$$\neg defend(A) \text{ at } t = 4 \quad \models B_{D,4}^{0.9,\ 1}(Obs_D(exec(A)) \vee Obs_D(exec(B))).$$

If the defender chooses not to take any defensive action, the intruder in turn is unable to distinguish between the situations where the defender deliberately took no action and where the defender simply missed the attack, i.e., the defender considers all threads possible that contain node 6 or node 4. Thus, the intruder has the following belief in actually being able to execute malicious code on a target system undetectedly:

$$\neg Obs_I(defend(A)) \text{ at } t = 4 \quad \models B_{I,4}^{0,\ 0.2}(\phi),$$

$$\text{with } \phi = \neg Obs_D(exec(A)) \wedge \neg Obs_D(exec(B)) \wedge (exec(A) \vee exec(B))$$

Since both the intruder and defender know the possible attack sequences, it is also possible to analyze belief states of the respective opponent: In the considered situation, the defender knows that the attacker has not observed any defense action and can therefore not distinguish between nodes 4 and 6. However, since the defender was not able to rule out a previous attack on B, from his point of view the intruder could still also consider nodes as 83, 85, 94, and 96 as possible (these are the nodes where B was attacked before and no defend action was taken). Still, the defender has a rather high belief in the intruder's actual belief state, as expressed in the following nested belief:

$$B_{D,4}^{0.8,\ 1}(B_{I,4}^{0.0,\ 0.2}(\phi))$$

5 Conclusion

In this paper we proposed a well-defined theory to formalize multi-agent beliefs in a security context. This formal representation enables the analysis of the adversary's belief evolution depending on specific actions. At a minimalistic example we demonstrated how a formal belief state analysis can be carried out. Next to formal representations of both the intruder's and defender's beliefs, this is especially useful to gain an opportunity to reason about nested beliefs. This provides novel means of assessing the expected utility of any action when planning a defense strategy: Along with analyzing the direct effect of any action on the network, we can also analyze how any action will influence the belief state of the opponent. With the use of more sophisticated attack models, this enables the defender to drive the intruder into desired safe states, where the intruder expects to achieve her goal, but is actually unable to cause real harm.

We only chose a minimalistic example because a manual analysis in complex attack graphs becomes infeasible by hand due to an excess amount of possible worlds. Still, PDT logic performs well in those and does not limit our approach. Future work includes a description of an—currently in implementation—autonomous system for analysis of complex attack graphs and an experimental evaluation of the demonstrated profound theoretic approach in complex settings.

References

1. Brdiczka, O., Liu, J., Price, B., Shen, J., Patil, A., Chow, R., Bart, E., Ducheneaut, N.: Proactive insider threat detection through graph learning and psychological context. In: Security and Privacy Workshops (SPW). pp. 142–149. IEEE (2012)
2. Chiesa, R.: Peering in the soul of hackers: HPP (the hacker's profiling project) v2.0 reloaded. In: 8.8 Security Conference, Santiago, Chile. 8dot8 (2012)
3. Chiesa, R., Ducci, S., Ciappi, S.: Profiling hackers: the science of criminal profiling as applied to the world of hacking. CRC Press (2008)
4. Ingols, K., Lippmann, R., Piwowarski, K.: Practical attack graph generation for network defense. In: Computer Security Applications Conference. pp. 121–130. IEEE (2006)
5. Jha, S., Sheyner, O., Wing, J.: Two formal analyses of attack graphs. In: Computer Security Foundations Workshop. pp. 49–63. IEEE (2002)
6. Lippmann, R., Ingols, K., Scott, C., Piwowarski, K., Kratkiewicz, K., Artz, M., Cunningham, R.: Validating and restoring defense in depth using attack graphs. In: Military Communications Conference (MILCOM). pp. 1–10. IEEE (2006)
7. Martiny, K., Möller, R.: A probabilistic doxastic temporal logic for reasoning about beliefs in multi-agent systems. In: 7th International Conference on Agents and Artificial Intelligence (ICAART) (2015)
8. Noel, S., Jajodia, S.: Optimal IDS sensor placement and alert prioritization using attack graphs. Journal of Network and Systems Management 16(3), 259–275 (2008)
9. Ou, X., Govindavajhala, S., Appel, A.W.: Mulval: A logic-based network security analyzer. In: USENIX Security (2005)
10. Ou, X., Singhal, A.: Attack graph techniques. In: Quantitative Security Risk Assessment of Enterprise Networks, pp. 5–8. Springer (2011)
11. Phillips, C., Swiler, L.: A graph-based system for network-vulnerability analysis. In: Workshop on New security paradigms. pp. 71–79. ACM (1998)
12. Rogers, M.K.: A social learning theory and moral disengagement analysis of criminal computer behavior: An exploratory study. Ph.D. thesis, University of Manitoba (2001)
13. Roy, A., Kim, D.S., Trivedi, K.: Cyber security analysis using attack countermeasure trees. In: 6th Annual Workshop on Cyber Security and Information Intelligence Research. p. 28. ACM (2010)
14. Roy, A., Kim, D.S., Trivedi, K.: Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees. In: Dependable Systems and Networks. pp. 1–12. IEEE (2012)
15. Sommestad, T., Ekstedt, M., Johnson, P.: Cyber security risks assessment with bayesian defense graphs and architectural models. In: 42nd Hawaii International Conference on System Sciences. pp. 1–10. IEEE (2009)
16. Theoharidou, M., Kokolakis, S., Karyda, M., Kiountouzis, E.: The insider threat to information systems and the effectiveness of ISO17799. Computers & Security 24(6), 472–484 (2005)