Validation of IT Risk Assessments with Markov Logic Networks

Janno von Stülpnagel and Willy Chen

Softplant GmbH
Agnes-Pockels-Bogen 1
80992 Munich, Germany
{janno.stuelpnagel,willy.chen}@softplant.de
http://www.softplant.de/

Abstract. *Risk assessments of big and complex IT infrastructures comprise numerous qualitative risk estimations for infrastructure assets. Qualitative risk estimations, however, are subjective and thus prone to errors. We present an approach to detect anomalies in the result of risk assessments by considering information about inter-dependencies between various building blocks of IT landscapes from enterprise architecture management. We therefore integrate data from enterprise architecture and risk estimations using Semantic Web technologies and formalize common anomalies such as inconsistent estimations of dependent infrastructure components. To reflect the uncertainty of qualitative analyses we utilize Markov logic networks (MLN) to validate the resulting model and determine more probable and consistent estimations.*

Keywords: IT Risk Management · Markov Logic Networks · Risk Assessment · Validation · Semantic Web · OWL2-QL

1 Introduction

IT is a critical factor for modern enterprises. Therefore the business risks resulting from the usage of IT have to be managed in an appropriate manner. IT risk management is a systematic approach concerned with the identification of threats and vulnerabilities as well as the initiation of countermeasures for security risks. Risk assessment, a critical task within IT risk management, focuses on identifying threats for components in an IT infrastructure and estimating the risk level. These assessments are primarily based on qualitative expert opinions [1]. Despite the size and complexity of IT infrastructures - especially in large enterprises - the risks of new threats have to be assessed continuously and promptly [2]. Also, appropriate countermeasures have to be taken.

A centralized approach for risk assessment, where this job is carried out by a single organization unit with limited personnel or even a single person, is obviously not practical for large enterprises. A collaborative risk assessment approach, where different persons assess the risks for their domains and the results are later combined, can address the challenge of the size of assessed IT infrastructures. In such an approach multiple persons would independently make risk

assessments for the parts of the IT infrastructure, which are in their responsibility. All individual risk assessments as well as their results are afterwards combined into an overall assessment.

Such a collaborative approach, however, leads to new challenges. Domain experts with different backgrounds may have different understandings of threats and risks. Especially if inter-connected IT infrastructure components are evaluated by different persons, it is possible that assessments lead to differing or even contradictory results. Also, data from different points of time may result in inconsistent threats ratings.

Our approach to this challenge is to check the validity of the risk assessments and their results. We thereby utilize Semantic Web technologies to integrate risk assessments results with actual information about the IT landscape itself and Markov Logic networks (MLN) to reflect the probabilistic approach of IT risk management.

The paper is structured as follows. Section 2 briefly gives an overview risk assessment, Semantic Web technologies and Markov logic networks. In Section 3 we present our approach for validating risk assessment results by using Semantic Web technologies and Markov logic networks. We show the validity of our solution in a case study in Section 4 and analyze related works in Section 5. We conclude the paper by discussing our work and pointing out the future direction of our work.

2 Background

2.1 IT Risk Management and Risk Assessment

Following the definition of the European Union Agency for Network and Information Security (ENISA)[3], IT risk management has five main processes (see also Figure 1):

- Definition of Scope: within this process the basic conditions and requirements are defined for the following processes.
- Risk Assessment: this process consists of three activities
  - Risk Identification: creating a list of possible threats
  - Risk Analysis: analyzing the risk level for threats estimating the likelihood and impact of a threat
  - Risk Evaluation: determining if a risk is acceptable or define a treatment priority.
- Risk Treatment: implementing measures against critical risks
- Risk Communication: ensuring the information exchange about risks between decision-makers and other stakeholders
- Monitor and Review: measuring the efficiency and effectiveness of all risk management process

A more complete overview can be found in survey of the risk assessment methods from the ENISA [3]. Within risk identification data sources such as the "IT-Grundschutz Katalog" [4] provides thorough knowledge about possible threats for IT infrastructure components and possible countermeasures.

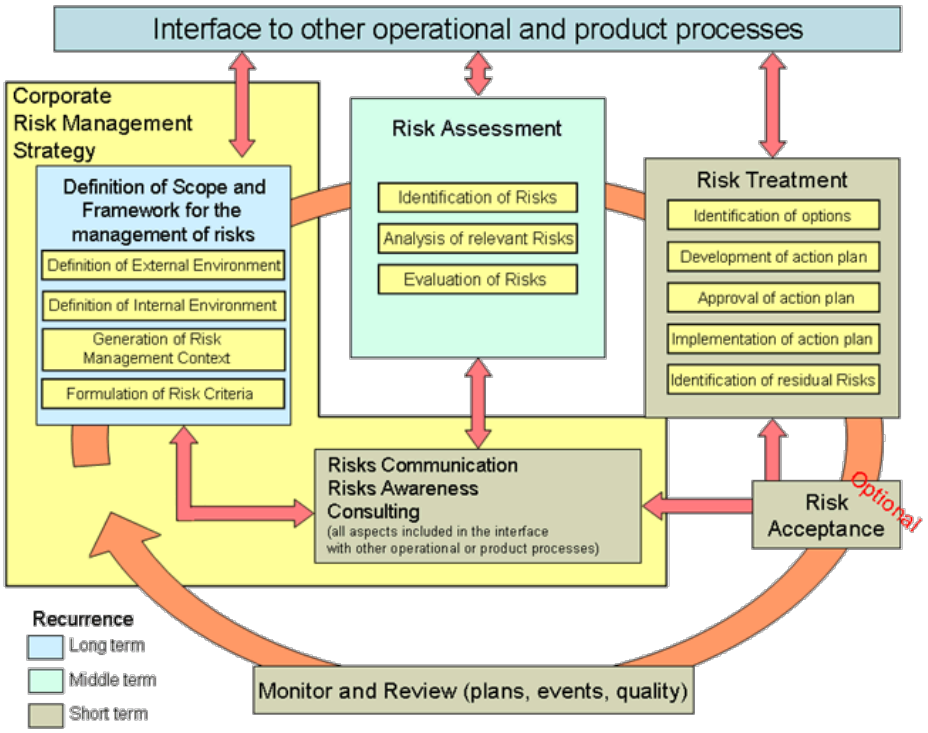

Fig. 1. Overall cycle of a Risk Management process after European Union Agency for Network and Information Security (ENISA) [3]

2.2 Semantic Web and Ontologies

The idea of Semantic Web is that data should be independent of it presentation and related to one another. This would allow for sharing and reusing data across applications and organization borders. A sound logical basis would make it possible to process the available data within the Internet [5]. Among others, ontologies have been defined as a core technology within the Semantic Web. Ontologies are formal specifications of terms within a specific domain and relations between those terms. They allow for making the semantics of data explicit and accessible to machines.

While the Semantic Web was initially intended as an extension of the World Wide Web, the technologies developed for it are also well-suited for integrating heterogeneous data in organizations. Semantic Web technologies have already been applied to a wide range of applications, from adding semantic metadata for sensors [6] and for enterprise architecture management (EAM) [7].

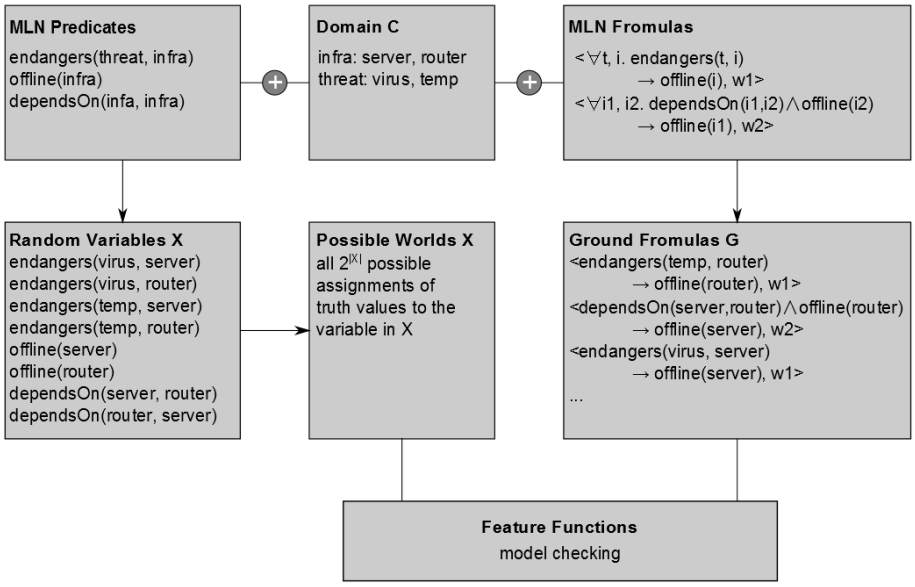

Fig. 2. The diagram describes the grounding of a Markov network. The grounded formulas G are generated by substituting each occurrence of every variable in the MLN Formulas with constants of the domain C. The possible worlds X are generated by giving all possible groundings of each predicate. Both the possible worlds X and the grounded formulas G are checked and provided with a value 1 if there are true and 0 otherwise. (Adapted from [8]).

2.3 Markov Logic Networks (MLN)

Markov logic networks (MLN) apply the idea of Markov Logics to first-order logic. MLN allows to assign weights to the formulas of the first-order logic[9] - i.e. specifying hard and soft formulas. Together with a set of constants, the MLN specifies a probability distribution over possible worlds.

Hard formulas are regular first-order formulas, which have to be fulfilled in every possible world. Soft formulas have weights that support worlds in which they are satisfied, but they do not need to, or in case of negative weights even should not be satisfied. The probability of a possible world, one that satisfies all hard formulas, is proportional to the exponential sum of the weights of the soft formulas that are satisfied in that world. This corresponds to the common representation of Markov networks as log-linear model [9].

An MLN is a template for constructing a Markov network. Figure 2 illustrates the structure of a MLN. For each set of constants, there is a different Markov network, following the rules given in the MLN. Markov logic makes the assumption that different constants refer to different objects (unique name assumption)

and the domain consists only of those constant and that no other objects (closed world assumption). An atom is a formula that consists of a single predicate and a grounding substitutes each occurrence of every variable in a formula with a constant. A set of grounded atoms is called a possible world.

An MLN L is a set of pairs $\langle F_i, w_i \rangle$, where F_i is a first-order logic formula and w_i is a real numbered weight [9]. The MLN L, combined with a finite set of constants $C = \{c_1, c_2, ...c_{|C|}\}$, defines a ground Markov network $M_{L,C}$ as follows:

1. $M_{L,C}$ has one binary node for each possible grounding of each predicate in L. The value of the node is 1 if the grounded atom is true and 0 otherwise.
2. $M_{L,C}$ contains one feature for each possible grounding of each formula F_i in L. The value of this feature 1 if the formula is true, and 0 otherwise. The weight of the feature is the w_i associated with F_i in L.

[9, p. 113]

Generally, a feature can be any real-valued function of the variables of the network. In this paper we use binary features, essentially making the value of the function equal to the truth value of the grounded atom.

The description as a log-linear model leads to the following definition for the probability distribution over possible worlds x for the Markov network $M_{L,C}$:

$$P(X = x) = \frac{1}{Z} exp\left(\sum_i w_i n_i(x)\right) \tag{1}$$

where Z is a normalization constant and $n_i(x)$ is the number of true groundings of F_i in x.

When describing the MLN we use the format $\langle$*first order formula*, *weight*$\rangle$. Hard formulas have infinite weights. If the weight is $+\infty$ the formula must always be true, if the weight is $-\infty$ it must always be false. A soft formula with weight 0 has equal probabilities for being satisfied in a world or not. Most query engines require the input to be split into two parts: the formulas (also called program) and the known evidence data.

There are two types of inference with Markov logic: maximum a posteriori (MAP) inference and marginal inference. MAP inference finds the most probable world given some evidence. Marginal inference computes the posteriori probability distribution over the values of all variables given some evidence.

3 Validation of IT Risk Assessments

Collaborative risk assessments in large IT infrastructures may be based on differing or even contradictory assessment for similar threats and dependent components. Thus the overall results may become inconsistent. In order to validate risk assessment we propose to utilize additional information about the IT infrastructure as well as knowledge about threats and their inter-dependencies.

Furthermore to reflect the probabilistic approach of risk assessments in a more precise manner we use probabilistic logics for doing the actual validation.

Our approach for the validation of IT risk assessment consists of three steps:

- **Formalizing IT infrastructure and risk assessment results**: create a formal model for risk assessment results and additional information of the IT infrastructure
- **Defining validation rules**: specify business rules to identify anomalies in the given model
- **Detecting anomalies**: utilize Markov Logic networks (MLN) to detect anomalies in given risk assessment results

3.1 Formalizing IT infrastructure and risk assessment results

IT risk assessment is considered with estimating the likelihood and impact of threats for IT components. Because neither threats nor IT components are completely isolated from each other, we are convinced that information about their dependencies provide a valuable source to validate risk assessment results. By aligning such information with the risk assessment results we can substantiate the actual assessments with an additional and independent source of knowledge. Especially if risk assessments are carried out collaboratively, not only the work load can be distributed but the assessments of all participants can also be validated against each other.

Enterprise architecture management (EAM), an IT management discipline, provides consolidated information about IT landscapes, the elements therein and their inter-dependencies starting from a business point of view to the actual infrastructure layer. This include information about servers, server locations and network connections within the IT infrastructure.

Catalog of threats, i.e. a list of possible threats for classes of IT components and dependencies between threats are a second relevant information source. "'IT-Grundschutz Katalog"' [4] provides a thoroughly list of possible threats for IT components. However, it does not include information about dependencies between threats. For example if there is a high likelihood for cable fire there must also be an increased likelihood for fire in the server room.

To integrate both information sources with the actual risk assessment results we create a formal model using ontologies. We therefore build on an ontology for an enterprise architecture[7], which we extend for this approach. We add appropriate details for covering detailed information about the IT infrastructure and include the "IT-Grundschutz Katalog" [4]. To provide an initial set of dependencies between threats we have manually added a number of dependencies. Finally we formalize information elements for risk assessment results. The resulting infrastructure ontology covers information on IT components of the IT infrastructure, possible known threats for standard IT components as well as several threat probabilities as results of risk assessments.

The ontology has been created by using the Web Ontology Language (OWL), which has been standardized by the World Wide Web Consortium (W3C) as an

ontology language for the Semantic Web. In particular we choose the QL language profile from the OWL 2 standard, which is optimized for "application that use very large volumes of instance data, and where query answering is the most important reasoning task" [10]. OWL 2 QL is based on the Description Logic DL-Lite$_R$, a decidable fragment of first-order logic, and allows for sound and complete conjunctive query answering in LOGSPACE with respect to the size of instance data. Using this technology provides on the one hand a means to capture and integrate relevant information about IT components and risk assessments. On the other hand such a formalization allows for additional approaches such as validating the risk assessment results using Markov Logic Networks (MLN), which we explain in more detail in the following sections.

3.2 Defining validation rules

Having information about IT components, threats and likelihood of threats within a single information base raises the question how to identify anomalies - especially focusing on the likelihood of threats, i.e. the results of risk assessments. Our approach is to define logical rules for validation. These rules reflect available knowledge about possible dependencies, which exist between two IT components and have to be considered appropriate when estimating the likelihood of threats.

For a very first proof of concept, we have identified following dependencies:

- an IT infrastructure component with a high probability for the threat such as "fire" should affect the same threat for all other IT infrastructure components in the same location
- a threat such as "unauthorized access to IT systems" or "malware" should affect other IT infrastructure components in the same network

We can express these dependencies using two universal formulas in the following first-order logic formulas:

$$\begin{aligned} &\texttt{hasProbability(infra1, threat1, prob1)} \wedge \\ &\texttt{hasLocalInfluence(threat1, threat2, prob2)} \wedge \\ &\texttt{inLocation(infra1, loc)} \wedge \texttt{inLocation(infra2, loc)} \Rightarrow \\ &\texttt{hasProbability(infra2, threat2, prob2)} \end{aligned} \tag{2a}$$

$$\begin{aligned} &\texttt{hasProbability(infra1, threat1, prob1)} \wedge \\ &\texttt{hasNetworkInfluence (threat1, threat2, prob2)} \wedge \\ &\texttt{inNetwork (infra1, net)} \wedge \texttt{inNetwork (infra2, net)} \Rightarrow \\ &\texttt{hasProbability(infra2, threat2, prob2)} \end{aligned} \tag{2b}$$

The formulas have 5 types of variables and 5 predicates. The different types of variables are `infra`[1] as the type of all infrastructure components, `threat` as the type of all threats, `net` as the type of all networks, `loc` as the type of

[1] In the following we use type writer fonts like `infra` for statements in first-order logic

all locations and `prob` as the type of all qualitative probability (like improbable, possible, probable ...). The predicate `hasProbability(infra1, thread1, prob1)` states that a IT infrastructure component `infra1` is endangerd by the threat `threat1` with the probability of occurrence of `prob1`. The predicates `inLocation(infra1, loc)` expressing that a IT infrastructure component `infra1` are located at `loc` and `inNetwork (infra1, net)` respectively that it is connected with the network `net`. `hasLocalInfluence(thread1, thread2, prob1)` and `hasNetworkInfluence (thread1, thread2, prob1)` are two predicates for expressing the local and network influence of a threat. The threat `thread1` influences the occurrence of the threat `thread2` with the probability of `prob1`.

The formula 2a states that if a threat endangers an IT infrastructure component and the threat has a local influence, another IT infrastructure component at the same location is endangered by the second threat with a probability specified in the influence. The second formula, formula 2b is analog to the first, only that it uses network influence instant of location influence. While these threat influences should be universal, the model is a simplification of the reality and it is quite possible that the expert did take some security measure into account which is not represented in our model. Therefore we will not use this as hard formulas, but as soft formulas.

3.3 Detecting anomalies

Finally, to utilize the collected information on the IT components, the threats and the risk assessment results together with the validation rules we have to translate them into a single MLN program and appropriate evidence.

We use the Formulas 2a and 2b with a weight of 1 and add the following formula:

$$\langle \texttt{hasInputProbability(infra, thread, prob)} \Rightarrow \texttt{hasProbability(infra, thread, prob)}, 5 \rangle \tag{3}$$

The formula 3 is used to give the result of the risk assertion occurrence probabilities in the evidence a weight, in this case 5, which states that the domain expert most likely did a correct assessment. This weight can be changed to express different confidentiality in the original risk assessment. It is also possible to extend our approached by adding extra predicates to express individual confidentiality. Our full MLN program can be found in appendix A.

The information in the ontologies can easily be translated into the MLN evidence by using a straightforward rewriting. Examples for MLN evidence can be found in section 4 in table 1, 2 & 3.

By using the marginal inference of the MLN, we can calculate the probability of every combination of the `hasProbability` predicate. We then can determine the probability of a threat occurrence on a specific IT infrastructure, by searching the highest `hasProbability` for this IT infrastructure component and threat. If the threat occurrence probability differs from the input then is a possible anomaly and should be further review by the experts.

4 Case Study

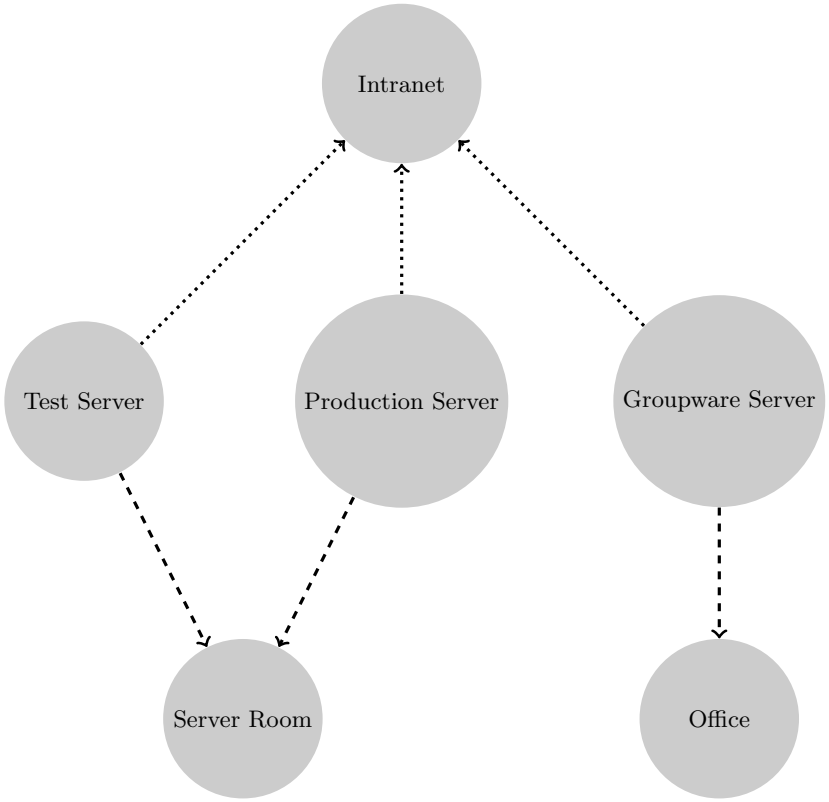

Fig. 3. The IT infrastructure for our case study. Dotted lines represent a connection to a network and dashed lines represent that a component stands in a location.

We demonstrate the validity our solution with a small use case. Because the creation of ontologies and mapping between them are already covered in other publications, for example [11] and [12], we focus on the MLN aspects. There are different MLN solvers available, e.g. Alchemy[2], Tuffy[3], and RockIt[4]. Within our work we use the RockIt Solver for our calculations. Our case study has three IT infrastructure components (`Test Server`, `Production Server` and `Groupware Server`) in two locations (`Server Room` and `Office`) and one network (`Intranet`) (see Figure 3). We use four occurrence possibility individuals in the following order: `Improbable`, `Possible`, `Probable` and `Very Probable`.

[2] http://alchemy.cs.washington.edu
[3] http://hazy.cs.wisc.edu/hazy/tuffy
[4] https://code.google.com/p/rockit

Table 1. Evidence Part 1: The IT infrastructure as MLN evidence

```
inLocation("Test Server", "Server Room")
inLocation("Production Server", "Server Room")
inLocation("Groupware Server", "Office")
inNetwork("Test Server", "Intranet")
inNetwork("Production Server", "Intranet")
inNetwork("Groupware Server", "Intranet")
```

Table 2. Evidence Part 2: The result of the risk assessment, which we want to check for invalidity

```
hasInputProbability("Test Server", "Fire", "Possible")
hasInputProbability("Test Server", "Unauthorized Access to IT Systems",
"Very Probable")
hasInputProbability("Production Server", "Fire", "Improbable")
hasInputProbability("Production Server", "Water", "Probable")
hasInputProbability("Production Server", "Unauthorized Access to IT Systems",
"Improbable")
hasInputProbability("Groupware Server", "Fire", "Improbable")
hasInputProbability("Groupware Server", "Unauthorized Access to IT Systems",
"Possible")
```

There are three threats to our IT infrastructure: `Fire`, `Water` and `Unauthorized Access to IT Systems`. The evidence for our MLN program has three parts. The first part is the IT infrastructure and the corresponding evidence is listed in Table 1. The second part is the risk assessment we want to check, which is given in Table 2. The last evidence part, see Table 3, defines the influence of the threat occurrence on other threats. We use the MLN program, see Appendix A for the complete program, to calculate the marginal inference.

The Table 4 lists a section of the result, only for the infrastructure `Test Server` and the threat `Fire`. Because `Possible` has the highest probability, with 0.999730, we take it as recommendation from the MLN. Because there is a big gab to the next highest probability, we can assume that the program is very confident in its recommendation. Table 5 lists the result, for each IT infrastructure/threat combination only the occurrence probability with the highest result is given. In five cases the result of the calculation is different from the input, which indicates an anomaly. The possibility for the threat `Unauthorized Access to IT Systems` changed in all three IT infrastructure components to `Probable`, which is logical because all three components are in the same network and hacking attacks often propagate through a network. The `Test Server` and `Groupware Server` got both a new `Water` threat with an `Improbable` probability, because both components have a fire threat. If there is the threat of a fire in a location then it is possible that the extinguishing of a fire results into a

Table 3. Evidence Part 3: The influence of the threats on each other

```
hasLocalInfluence("Fire", "Fire", "Possible")
hasLocalInfluence("Fire", "Water", "Improbable")
hasNetworkInfluence("Unauthorized Access to IT Systems",
"Unauthorized Access to IT Systems", "Probable")
```

Table 4. The Example of the result of our calculation for the infrastructure `Test Server` and the threat `Fire`

Component	Threat	Occurrence Probability	Marginal Inference Result
`Test Server`	`Fire`	`Possible`	0.999730
`Test Server`	`Fire`	`Very Probable`	0.492144
`Test Server`	`Fire`	`Probable`	0.491708
`Test Server`	`Fire`	`Improbable`	0.490258

water damage. The calculation gives thereby good indicators for invalidity and provides adequate correction recommendations.

5 Related Work

There are other approaches than MLN to combine logic and probability (see [13] for an overview). Because Bayesian networks already used for risk analysis [14], we will limit the discussion of related work to Bayesian logic programs (BLP) [15]. BLP unifies Bayesian networks with logic programming. While MLNs infers all probabilities from a given set of weights, BLP computes the probability for one query from the known probabilities in the resolution to the query statement. MLN has also the advantage that it uses undirected graphs. This has the effect that changes of one weight influence the whole graph and the results. Thereby it is more likely to find new relationships that where not explicitly modeled. This has on the other hand the disadvantage that it is harder to isolate a single variable. MLNs have been used for a wide range of problems, for instance in data integration [16] and determine the availability of IT infrastructure components[17].

While there are already approaches for finding errors in ontologies [18], there is more work needed to make these methods applicable in practice [19]. They are also limited by the expressiveness of the ontology language; therefore it is not possible to use soft formulas as indicators for finding errors. The combination of Semantic Web and Markov Logic Networks is also nothing new. It has already been used for link prediction and collective classification, for filling in missing attributes and relationships; entity resolution, for matching equivalent entities that have different names; information extraction, for adding structure to raw or semi-structured text [20].

Table 5. The results of our calculation, only the most probable threat occurrence probability is given for each infrastructure threat combination.

Component	Threat	Input Probability	Highest Marginal Inference Probability
Test Server	Unauthorized Access to IT Systems	Very Probable	Probable
Test Server	Fire	Possible	Possible
Test Server	Water	-	Improbable
Production Server	Unauthorized Access to IT Systems	Improbable	Probable
Production Server	Fire	Improbable	Improbable
Production Server	Water	Probable	Probable
Groupware Server	Unauthorized Access to IT Systems	Possible	Probable
Groupware Server	Fire	Improbable	Improbable
Groupware Server	Water	-	Improbable

6 Discussion

To our knowledge this paper presents a novel solution for finding anomalies in risk assessments. We utilize Semantic Web technologies to formalize different relevant data sources. Markov logic networks allows for detecting contradictory assessments and compute corrections. We have defined rules to validate the probability of threat occurrences and specified additional information sources relevant to the validation. We have shown the validity of our approach in a small case study.

Our solution has following advantages. By using Semantic Web technologies, i.e. an ontology, we describe the IT all relevant information elements and their dependencies and integrate data from EAM and risk assessments into such information graph. Such an information graph provides a solid foundation for analysis but also for additional means of support such a the automated validation of risk assessments.

We compute the marginal inference of the MLN model to get the probability distribution of all the variables. Comparing them with the actual risk assessment results leads to possible anomalies in the estimation of likelihood of threats. Moreover, by using MLN we can not only detect possible anomalies but also computer the most probable value. While it would be possible to use MAP for our approach, it would only provide the most probable world and no information about how probable other alternatives are.

The used rules for threat and location dependencies furthermore can not only be given as hard formulas but also as soft formulas. Such soft rules reflect the knowledge of domain experts in a more realistic manner. The MLN model also

allows for circular dependencies within components of the IT infrastructure and threats, which are quite common in real world scenarios. The weights used with the soft formulas can also be determined automatically using existing learning approaches [9]. It is therefore possible to iterative train the weights of the MLN by accepting or rejecting the suggested changes of the MLN calculation.

There are some limitations to our solution. The modeling with MLN is not as straight forward as the simple representation suggests [8]. The relationship between weights and the probabilities of the marginal inference result can be counterintuitive. The probability depends on the whole MLN, the program and the evidence, and even the number of individual can have an effect. The weight of a soft formula therefore does not directly correspond to a specific universal probability for this formula. Changing the weights in one formula shifts the relative weights of the possible worlds where the formula is true. This can affect the probability of variables which are not directly connected to the formula. This is the reason why we don't express the threat occurrence probability directly with weights but with constants. If we express the threat occurrence probability with weights, this would result very many weights. These would affect each other and it would be very hard to find the correct corresponding weights for the threat occurrence probability.

The scalability of our approach needs to be further investigated. While the OWL2-QL profile has a very good scalability, we need to test the MLN inference with bigger data sets. The marginal inference with MLN has the advantage that it uses a Gibbs sampling approach. The Gibbs sampling gets more accurate with each sample and it allows us thereby to choose how much time to invest into the accuracy of the probabilities.

Our approach does not use the order of the threat occurrence probability. It returns the probability with is supported by the most weights, but it does not find compromise solutions. If there is weight for a high and a low probability our approach returns the one with more weight and not a medium probability.

7 Conclusion

In this paper we have described an approach to find anomalies in collaborative risk assessments. Our solution aims at improving the confidence in collaborative risk assessments by detecting contradictory assessments of threats for same or dependent infrastructure components. First, we combine and integrate relevant information about the IT infrastructure itself, threats and their interdependencies as well as various risk assessments into a single formal model. We therefore utilize Semantic Web technologies which allows for the integration of heterogeneous data sources. We finally translate the resulting model into a Markov Logic network and compute the marginal inference to detect anomalies in risk assessments and to determine more probable assessments for risks.

While we have focused our work in this paper primarily on the probability of threat occurrences, future work may also consider threat impact and criticality. Also, we plan to research the use of additional confidence values that may be

collected from the assessors during the identification of risks. This would provide us with an additional source for aligning collaborative risk assessments.

Acknowledgement

This work has been partially supported by the German Federal Ministry of Economics and Technology (BMWI) in the framework of the Central Innovation Program SME (Zentrales Innovationsprogramm Mittelstand - ZIM) within the project "Risk management tool for complex IT infrastructures".

References

1. Zambon, E., Etalle, S., Wieringa, R.J., Hartel, P.: Model-based qualitative risk assessment for availability of it infrastructures. Software & Systems Modeling **10**(4) (2011) 553–580
2. IBM X-Force: Mid-year trend and risk report 2013. Technical report, IBM X-Force (2013)
3. Technical Department of ENISA Section Risk Management: ENISA: Risk management: implementation principles and inventories for risk management/risk assessment methods and tools. Technical report, European Network and Information Security Agency (ENISA) (2006)
4. Bundesamt für Sicherheit in der Informationstechnik: IT-Grundschutz-Kataloge. Technical report, Bundesamt für Sicherheit in der Informationstechnik (2013)
5. Berners-Lee, T., Hendler, J., Lassila, O., et al.: The semantic web. Scientific american **284**(5) (2001) 28–37
6. Sheth, A., Henson, C., Sahoo, S.S.: Semantic sensor web. Internet Computing, IEEE **12**(4) (2008) 78–83
7. Chen, W., Hess, C., Langermeier, M., Stülpnagel, J., Diefenthaler, P.: Semantic enterprise architecture management. In: ICEIS (3). (2013) 318–325
8. Jain, D.: Knowledge engineering with markov logic networks: A review. In Beierle, C., Kern-Isberner, G., eds.: Evolving Knowledge in Theory and Applications. 3rd Workshop on Dynamics of Knowledge and Belief (DKB-2011) at the 34th Annual German Conference on Artificial Intelligence, KI-2011, Berlin, Germany, October 4, 2011. Proceedings. Volume 361 of Informatik-Bericht., Fakultät für Mathematik und Informatik, FernUniversität in Hagen (2011) 16–30
9. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning **62**(1-2) (2006) 107–136
10. Motik, B., Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 web ontology language: Profiles. W3C recommendation (2012)
11. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The knowledge engineering review **18**(01) (2003) 1–31
12. Braun, S., Schmidt, A., Walter, A., Nagypal, G., Zacharias, V.: Ontology maturing: a collaborative web 2.0 approach to ontology engineering. In Noy, N., Alani, H., Stumme, G., Mika, P., Sure, Y., Vrandecic, D., eds.: Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge (CKC 2007) at the 16th International World Wide Web Conference (WWW2007) Banff, Canada, May 8, 2007. Volume 273 of CEUR Workshop Proceedings. (2007)

13. Braz, R., Amir, E., Roth, D.: A survey of first-order probabilistic models. In Holmes, D., Jain, L., eds.: Innovations in Bayesian Networks. Volume 156 of Studies in Computational Intelligence. Springer Berlin Heidelberg (2008) 289–317
14. Weber, P., Medina-Oliva, G., Simon, C., Iung, B.: Overview on bayesian networks applications for dependability, risk analysis and maintenance areas. Engineering Applications of Artificial Intelligence **25**(4) (2012) 671–682
15. Kersting, K., De Raedt, L.: Bayesian logic programs. CoRR **cs.AI/0111058** (2001)
16. Niepert, M., Noessner, J., Meilicke, C., Stuckenschmidt, H.: Probabilistic-logical web data integration. In Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P., eds.: Reasoning Web. Semantic Technologies for the Web of Data. Volume 6848 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2011) 504–533
17. von Stülpnagel, J., Ortmann, J., Schoenfisch, J.: It risk management with markov logic networks. In: Advanced Information Systems Engineering, Springer (2014) 301–315
18. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging owl ontologies. In: Proceedings of the 14th international conference on World Wide Web, ACM (2005) 633–640
19. Stuckenschmidt, H.: Debugging owl ontologies-a reality check. In: EON. (2008)
20. Domingos, P., Lowd, D., Kok, S., Poon, H., Richardson, M., Singla, P.: Uncertainty reasoning for the semantic web i. Springer-Verlag, Berlin, Heidelberg (2008) 1–25
21. Noessner, J., Niepert, M., Stuckenschmidt, H.: Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models. In des Jardins, M., Littman, M., eds.: Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2012, Bellevue, Washington, USA, AAAI Press (2013)

A RockIt MLN

For our MLN, we use the syntax of RockIt [21]. RockIt expects first order formulas in conjunctive normal form (CNF). An online version of RockIt is available here: `http://executor.informatik.uni-mannheim.de/systems/rockit/`

```
*inNetwork(infra, net)
*inLocation(infra, loc)
*hasInputProbability(infra, thread, prob)
hasProbability(infra, thread, prob)
*hasLocalInfluence(thread, thread, prob)
*hasNetworkInfluence(thread, thread, prob)

5 !hasInputProbability(infra, thread, prob) v
  hasProbability(infra, thread, prob)

1 !hasProbability(infra1, thread1, prob1) v
  !hasLocalInfluence(thread1, thread2, prob2) v
  !inLocation(infra1, loc) v !inLocation(infra2,  loc) v
  hasProbability(infra2, thread2, prob2)

1 !hasProbability(infra1, thread1, prob1) v
```

```
!hasNetworkInfluence(thread1, thread2, prob2) v
!inNetwork(infra1, net) v !inNetwork(infra2, net) v
hasProbability(infra2, thread2, prob2)
```