Argumentation-Based Security Requirements Elicitation: The Next Round

Dan Ionita
Services, Cybersecurity and Safety Research Group, University of Twente, The Netherlands
D.Ionita@utwente.nl

Jan-Willem Bullee
Services, Cybersecurity and Safety Research Group, University of Twente, The Netherlands
J.H.Bullee@utwente.nl

Roel J. Wieringa
Services, Cybersecurity and Safety Research Group, University of Twente, The Netherlands
R.J.Wieringa@utwente.nl

***Abstract*—Information Security Risk Assessment can be viewed as part of requirements engineering because it is used to translate security goals into security requirements, where security requirements are the desired system properties that mitigate threats to security goals.**

To improve the defensibility of these mitigations, several researchers have attempted to base risk assessment on argumentation structures. However, none of these approaches have so far been scalable or usable in real-world risk assessments.

In this paper, we present the results from our search for a scalable argumentation-based information security RA method. We start from previous work on both formal argumentation frameworks and informal argument structuring and try to find a promising middle ground. An initial prototype using spreadsheets is validated and iteratively improved via several Case Studies. Challenges such as scalability, quantify-ability, ease of use, and relation to existing work in parallel fields are discussed. Finally, we explore the scope and applicability of our approach with regard to various classes of Information Systems while also drawing more general conclusions on the role of argumentation in security.

I. Introduction

Information security Risk Assessments (RA) are brainstorming sessions during which a group of experts look at an infrastructure and try to find and rank vulnerabilities of the system, while proposing countermeasures that could mitigate the most dangerous of these so as to achieve an acceptable level of risk. We view these countermeasures as security requirements of the target system. As such, the process leading up to them can be framed as Security Requirements Engineering.

Risk is commonly evaluated as a product of the likelihood and the impact of an undesirable event (such as an attack). Often though, these cannot be estimated quantitatively, for example because the event is very rare. Some researchers have proposed an alternative approach to RA: producing informal but structured arguments that do not require quantification but still allow the identification of important risks while also capturing the rationale behind mitigation decisions. Such justifications can serve the purpose of prioritizing countermeasures and providing traceability for security requirements and the resultant investments. Furthermore, they can serve as a basis for future assessments.

Security checklists, traditionally used to support Risk Assessments, embody experience from the past, and are good at both detecting and mitigating known risks in an effective manner. However, problems arise when these are used for assessing new or changing architectures or for predicting novel risks. For example, the growing complexity of networked applications and critical infrastructures often makes checklists less useful.

A. Goal

We aim to replace or extend such checklists by a more formal argumentation-based tool to support such brainstorming sessions while providing better trace-ability of the decisions made and allowing (semi-)automated reasoning. A first step in this direction was the research conducted with Prakken [1]. This turned out to require too much bookkeeping overhead for common application scenarios.

Thus, this paper aims to provide argumentation support for Risk Assessment that is both scalable and usable in practice in the sense that it does not make unrealistic assumptions about quantification and does not employ complex argumentation structures. Furthermore, such argumentation support must be adaptive to changes in the architecture or to new vulnerabilities.

II. Related Work

The research described in this paper was conducted in the context of the TREsPASS[2] Information Security project whose goal is to develop a RA methodology which can cope with the fact that most numbers are not available or inaccurate.

A. Informal Argument Structures in Security

The starting point for most work on argumentation is Toulmin's argument structure [3], the rhetorical applications of which were later highlighted in [4].

Although this was inspired by legal arguments, its flexibility meant that it was also applicable to other fields, including Information Security. Most notably, Haley et al. [5] were one of the first to use Toulmin's argument structure for showing that security requirements satisfy security goals. They also described a process for eliciting such requirements by supporting formal proofs with informal arguments [6], later integrated into a framework for representing and analysing

978-1-4799-6339-3/14/$31.00 © 2014 IEEE

ESPRE 2014, Karlskrona, Sweden

Security Requirements [7]. However, so far the technique is unable to deal with some practical constraints, such as incomplete information, uncertainty and limited resources available for risk assessment and mitigation.

A second iteration of Haley's approach, focused on the elicitation of such requirements via a risk assessment, was proposed by Franqueira et al [8]. It made use of public catalogues in order to support the identification of rebuttals and mitigations. It also introduced defeasibility relationships between the arguments. However, this method was only validated on a small scale example and, due to its intrinsic complexity, we expect scalability to be a problem if an attempt were made to apply it to full-fledged Information Systems.

Later, a tool called OpenArgue[9] was developed to support the process, which attempted to display graphical representations of both the internal and external argument structures. However, using this tool requires writing pseudo-code that describes the argument. This imposes significant overhead for the risk assessors and implies knowledge of the syntax. Furthermore, the visual representations quickly explode in size when realistic sets of arguments are loaded, making them only slightly more understandable than the textual version.

B. Formal Argumentation Frameworks

Recently, attempts have been made to use logical formalisms in the external structure of the arguments in order to better describe the relationships between them and to allow various formal analyses and reasoning. One of the most prominent attempts is the ASPIC Argumentation Framework[10].

Based on this generic framework, together with Prakken [1], we devised an argumentation approach tailored for Risk Assessment. It replaces Toulmin arguments with ASPIC arguments in an attempt to formalize the process as an argumentation game in which assessors exchange arguments about how the system can be attacked and which countermeasures are feasible. The game is dynamic, as the defenders can add or remove elements from the target architecture as the game progresses. While this approach achieved good traceability, due to its high level of formalism it is very hard to use: all arguments have to be defined with regard to a knowledge-base using a strict syntax. While the concept of an argumentation game seems promising, the high overhead added by the formal logic framework poses significant threats to both the scalability and usability of the approach.

III. Approach

A. Starting Point

Our current proposal, similar to Prakken's [1] is centred around an argumentation game. We stripped down most of the logical formalism that was part of the ASPIC framework, while keeping only the basic inter-argument structure and elements of the work-flow. The overall structure is still that of an argumentation game, but without a pre-defined set of explicit inference rules or ranking. We rather let these emerge during the game.

Arguments are defined with regard to a (simplified) Toulmin structure. In this sense, we are attempting to find a middle ground between the Prakken et al. [1] approach and the Toulmin-based approaches ([7], [8], [9]).

A major distinction from previous approaches is the use of a simple tabular format as both input and storage of arguments where these are entered from top to bottom as they are introduced along the game (see Table I).

However, unlike Prakken[1], we do not differentiate between different types of counter arguments and assume the inference between fact plus assumptions and the claim to be implied.

Compared to OpenArgue, the main difference lies in the introduction of a semi-formal defeasibility relationship between arguments that allows semi-automated reasoning as well as the computing of several types of reports or summaries. Furthermore, we introduce a way by which components can be referenced so that potentially conflicting arguments can be highlighted.

B. Resulting Method

The assessors alternate between playing "defenders" and "attackers". Each "team" then takes turns formulating arguments either for or against the security of the system. In Table I, each row represents such a turn and describes one argument, starting with the attackers. Attacker arguments—marked by an A in the first column—describe Risks (in terms of possible attacks or vulnerabilities), while defender arguments describe ways to mitigate such Risks (by introducing countermeasures, transferring or accepting them).

There is both an internal and an external argument structure:

- Internally, each argument consists of three parts: a claim, one or more assumptions and one or more facts. Each part is given a unique ID. The facts are either physical facts, or known technical specifications of the target infrastructure. Assumptions are important parts of the argument that the assessors are not certain of. The claim is the core conclusion of the argument.
- Externally, there exist defeasibility relationships between arguments. That is, each argument can rebut (i.e. attack) one or more previous arguments by invalidating the claim itself or one of its assumptions. However, facts cannot be invalidated.

The two structures described above allow that each argument directly rebuts a part of any previous argument. The *Rebuts* column in Table I points to the ID of the part being rebutted. To represent the resulting states, we adopt part of Dung's abstract argumentation framework [11], in which each argument can at any moment in time be in one of two states: IN or OUT. Once an argument is successfully rebutted (that is, the opposing team proposes a valid counterargument), it becomes OUT, with the counterargument being IN. This can continue recursively, applying the following rules:

- An argument is IN if all its counterarguments are OUT. IN arguments have not been successfully defeated in the argument so far.

TABLE I
SNAPSHOT OF AN ARGUMENT GAME FOR CASE STUDY 1: HOME PAYMENTS SYSTEM

Player	Claim		Assumptions		Facts		Rebuts	Status	Flags
A/D	ID	STRING	ID	STRING	ID	STRING	ID	IN/OUT	T/R
A	C0	Listen in to Bluetooth: gather authentication or user data	A0	Bluetooth signal can be received outside	F0	Range of Bluetooth is 10m		OUT	
D	C1	Authentication data is encrypted	A1	AES encryption is good enough	F1	Bluetooth with 2.1 (AES) encryption	C0	IN	R
A	C2	User socially engineered to wire money	A2	Attacker can gain user's trust;	F2	–		OUT	
D	C3	Social Engineering is user risk	A3	–	F3	End-user agreement transferring liability for SocEng attacks	C2	IN	T
A	C4	User credentials can be stolen by peeking through the window	A4	Apartment located on bottom floor(s); Curtains open	F4	–		IN	

- An argument is OUT if it has a counterargument that is IN. OUT argument have been successfully defeated in the argument so far.

To test out the effectiveness and applicability of these ideas in case studies, we implemented the method as a spreadsheet containing underlying formulas for recursively determining the argument state, which is represented using colours (red for OUT, green for IN).

We initially assumed the following loose mapping from argument states to Risk states:

- Attacker arguments that are IN at the end of the game are *accepted* (retained) Risks (e.g. last row in Table I)
- Attacker arguments that are OUT at the end of the game are *Reduced*, totally *eliminated* (e.g. first two rows in Table I), or *transferred* (e.g. middle rows in Table I)

A secondary functionality is relating arguments to system architecture. The risk assessors start by drawing an architecture diagram of the Target of Assessment (and optionally its context), and enter a list of architecture components (nodes or connectors in the diagram) in the spreadsheet. Arguments are then automatically tagged with the labels of architecture components that they refer to as they are typed. This makes it easier for the assessors to identify potential conflicts in their statements as they are making them. Such a conflict occurs when a fact or statement is in contradiction with a previous fact or statement or if it is impossible to realize due to a previous statement about the same component. This labelling also helps avoid inconsistent views about the system among the assessors.

As stated above, facts cannot be invalidated, so it is important they are mutually consistent. Some facts may be properties of the Target of Assessment postulated by the defenders of the system. This means that we assume it is in the power of the defender to make decisions about the ToA architecture, and that these decisions will be implemented. For the system developers these facts are hence requirements.

IV. RESEARCH METHOD

Based on previous work, including a survey of common Risk Assessment methods, tools and frameworks [12], we tried to identify possible limitations or current approaches and scope for improvement.

To advance the state of the art, the approach was iteratively validated and improved via Technical Action Research and two Case Studies. This has resulted in four versions of the approach, each supported by spreadsheets:

1) Reduce complexity of ASPIC-based approach of Prakken et al[1] $\Rightarrow$ 1st version
2) Improve the method after a Case study on Home Payments System with students $\Rightarrow$ 2nd version
3) Improve the method after a Risk Assessment of the Home Payments System with experts $\Rightarrow$ 3rd version
4) Improve the method after a Case Study on Cloud-based Infrastructures $\Rightarrow$ 4th and latest version; it is this version that is described above.

The Case Studies are described in Sections V and VI.

V. CASE STUDY 1,2: THE HOME PAYMENTS SYSTEM

A. Case Description

This case study is centred on customer privacy protection and is owned by one of the project partners. The system consists of set-top boxes located in customer's homes, some centralized servers and personalized NFC-active bank cards. The set-top boxes are connected to the TV and allow the user to perform various financial operations (including but not limited to online banking, allocation of funds, payment of bills and online shopping) from the comfort of their home, by using the card as a means of authentication. A basic architecture diagram is shown in Figure 1.

This case study is intentionally under-specified for two reasons: (1) the system, developed by Consult Hyperion, is still at the prototype stage; and (2) this allows for more freedom with regard to the design decisions that can be taken

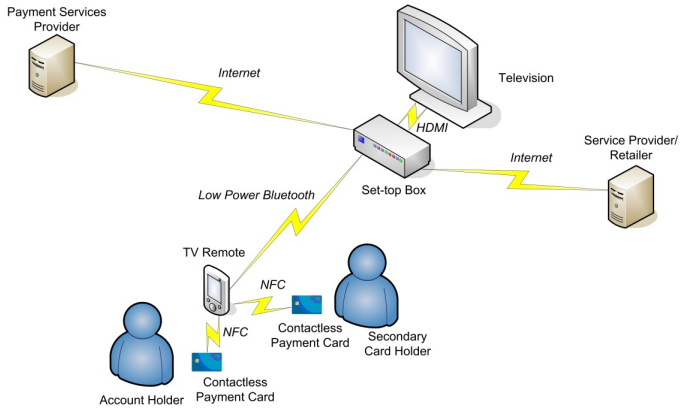

Fig. 1. Home Payments System

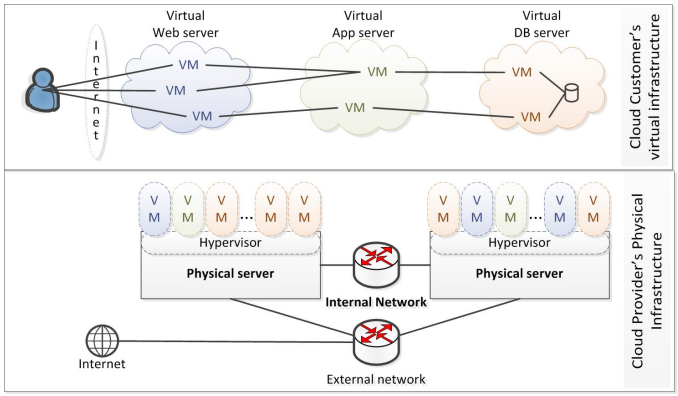

Fig. 2. IaaS Cloud architecture

during the assessment. Thus, we are also looking to test if the approach might be used during product design phases, where Security Requirements elicitation is more crucial than after implementation.

B. Case-Specific Observations

This case study was used twice: a pilot round in which the assessors were IT Security PhD students and a second round with senior Information Security researchers. The following observations were made during both sessions:

Assumptions are non-exhaustive: Attacker assumptions are usually about the system and its context, while the defenders usually make assumptions about the attacker's profile and skills. A common problem with assumptions is that even when asking the participants to make them explicit, there are always some that remain hidden. Hidden assumptions cannot be explicitly attacked. This can be overcome by stating an opposing assumption as part of the counter-argument.

Reduced Risks are not the same as eliminated Risks: Attacker arguments which are out OUT at the end of the game signify eliminated or reduced risks (e.g. middle rows in Table I). While for eliminated Risks, the attack is completely prevented, in the case of reduced risks, although the impact or likelihood have been sufficiently reduced, the attack itself might still be possible. To represent this, defender arguments that only partly mitigate the Risk (to an acceptable level) are flagged "R" (i.e. reduced).

Transferred risks should be clearly marked: Transferring Risks is a treatment option available during Risk Assessments (usually accomplished via insurance, end-user agreements, etc.). This means that the attack is still possible but liability for the potential negative consequences has been transferred. To support this, arguments that transfer the consequences of a Risk are flagged "T" (i.e. transferred).

Separate teams are better.: Allowing participants to take turns playing attacker and defender leads to them already having a counter-argument in mind when stating an argument, and thus subverts the argumentation dynamics of the game. Separate, fixed teams do not only mitigate this, but also instil a level of competitiveness between the two teams, resulting in better-formulated arguments.

VI. Case Study 3: The Cloud-Based Infrastructure

A. Case Description

Cloud-based implementations are now commonplace, with various service providers outsourcing their IT infrastructure to the cloud. Such virtualized infrastructures give rise to completely new categories of Risk, as well as new requirements with regard to identifying and mitigating such risks. As such, in order to explore the limits of applicability of the new method, a Risk Assessment was conducted in collaboration with IBM Research Zurich. As the target for assessment, a generic IaaS infrastructure was imagined. An overview of this infrastructure is presented in Figure 2.

The architecture consists of two infrastructure layers:

- A physical layer, owned by the Cloud Provider. This consists of some servers, connected to each other via an internal network, and to the Internet via an external uplink. Each server runs a number of virtual machines belonging to the Cloud Customers which are managed via an interface called a Hypervisor. The entire physical infrastructure is managed by the Cloud Administrator, who can also access and manage the virtual machines (e.g. resource allocation) via an SSH connection to the Hypervisor console.
- A virtual layer, consisting of a large number of virtual machines, networks and databases. Each Cloud Customer owns and controls a sub-set of virtual machines. A Customer Administrator is usually responsible for configuring and managing these for each Cloud Customer.

B. Case-Specific Observations

This case study was conducted with junior researchers on virtual infrastructures from IBM Zurich. The observations listed in the first two Case studies were confirmed during the third study. However, many new observations surfaced, mostly specific to Cloud-based infrastructures:

The introduction of a third, virtual, layer in-between the physical and logical domains, together with the dynamic nature

of this layer makes assessing Cloud-based infrastructure a more complex undertaking compared to traditional Information Systems. When you add to this the introduction of the cloud provider as a new stakeholder and the mixed ownership of components across these three domains, the separation between system and context fades away and the amount of incomplete or missing information w.r.t the infrastructure explodes.

As described in Cloud Risk Assessment documents from ENISA [13] and the CSA [14], the fact that cloud customers do not usually have any control of or information about the physical infrastructure and resource allocation itself gives rise to a new host of vulnerabilities, ranging from resource exhaustion to collocation exploits.

Because none of the stakeholders have the ability to directly influence the components owned or managed by the others, countermeasures for Cloud-specific Risks are mostly implemented via SLA clauses. These commonly have expiration dates and even time constraints for implementation due to contractual periods, making them significantly different to the more technical countermeasures the technique was designed to handle. This is because there is no clear transfer or mitigation of the Risk. Instead, partial transfer of risk by means of such SLA clauses is common. The way these clauses are written and how compensation is specified determines the degree to which Risk is transferred. The proliferation of organizational entities with heavy reliance on SLAs dictating the relationships between such entities also make assessments more complex, as well as making our method less applicable.

Despite the above difficulties, we were able to complete the Risk Assessment. However, the results looked significantly different from those identified in the previous case study. First of all, the attacker's facts were almost always missing. The same was true for the defender's assumptions. The attacker's assumptions usually implied the existence of a known vulnerability while the claims mostly referred to relationships between Vulnerabilities and Risks that are described in [13] and [14]. Therefore, it seems that while the method is flexible enough to be applied to such different scenarios, it does not offer significant added value in cases like this.

VII. Validity and Scope

In order to avoid problems related to participant bias, repeated testing and maturation, completely different panels of experts were used for each of the Risk Assessment sessions. None of the participants had seen or heard about the method before the session so as to further avoid selection bias. Only observations that have been confirmed in at least two of the three sessions are described in this paper.

Furthermore, subsequent iterations produced improved effects, with the exception of the assessment of Cloud-based infrastructure. This leads us to believe that the effects are indeed produced by the tool, although the tool has limited applicability when dealing with (partially) virtualized and/or outsourced infrastructures.

A. Applicability

During validation, and especially in the third case study, conclusions were drawn about the limitation of the method. The flexible argument structure and lack of Security-specific features in theory make it applicable to a wide range of scenarios that are based on brainstorming and require traceability.

However, we have noticed that for at least some of these cases it would not provide worthwhile utility. This is partly due to the significant overhead of formulating each argument according to the template. When the statements or decisions are simply a claim based on a few assumptions or facts, there is little added benefit in attempting to structure them. Such cases do not benefit from describing the defensibility relationship between various arguments because there is no back-and-fourth rhetorical discourse or the argument's inner workings are of little significance to the conclusions.

Finally, the added benefit of using argumentation is mostly visible when the architecture is known because to allow traceability to components (further explained in Section IX).

VIII. Future Work

As the features required go beyond what is normally achievable via spreadsheets, a dedicated software tool will most likely offer significant increases in the usability and scalability of the approach, while potentially adding extra functionality.

Besides improving the tool, we have identified several other directions for improvement and further research with regard to argumentation in Security.

We have observed that arguments, in the context of Risk Assessment and/or Security Requirements Elicitation, take the form of traceability links between the requirements, vulnerabilities, components and attacker profiles. In this respect, they outperform the use of checklists. However, the added overhead raises the question of what level of traceability is necessary and sufficient and how that level can be provided without overburdening the process. While the approach described in this paper come closer to this desired equilibrium than ASPIC and OpenArgue for some types of assessments, more research is required in order to more precisely determine at what level the method's cost are outweighed by its benefits..

Furthermore, due to limited time, only two real-world cases have been used for validation. To properly assess the potential benefits and limitations of the process described in this paper, as well as its wider applicability, more Case Studies would need to be conducted on other cases.

Finally, we have identified a relationship between our approach and current approaches in the field of design rationale and group decision support systems, described in Sections VIII-B and VIII-A respectively. Exploiting this similarity might not only expand the applicability and scope of the method, but could improve the tool itself by integrating some proven elements from these, more developed fields. Consequently, this link has to be further explored.

A. Relation to Group Decision Support Systems

In each round, only one argument is described. This means the team has to agree on the argument being presented to the other team before submitting it. This suggests a parallel with group decisions support systems (otherwise known as group decision rooms), where a software tool mitigates the interactions between various stakeholders in order to help achieve a consensual opinion. Such concepts could also be applied to our approach, thus increasing its (perceived) utility by providing the users with extra functionality.

B. Relation to Design Rationale

The Security Requirements resulting from the assessment could be in the form of technical countermeasures, but can also specify policies or more general design decisions w.r.t the target system. Especially for systems under development or prototypes, these Security Requirements together with the claims they support are very similar to "design rationale" in the sense that they describe and motivate the desired properties of the system; in this case, in terms of the Risks they are preventing.

The principles of capturing "design rationale as a by-product" of other related decision-making tasks, described in [15, Chapter 4.3] could be applied in order to evolve the approach in this direction as well as providing a more consistent method of storing the arguments.

IX. Conclusions

Most Case Study participants agreed that the method proposed in this paper might be feasible, provided that the obvious limitations of spreadsheet tables are effectively mitigated. That is, bookkeeping is reduced compared to the ASPIC-based approach and could be further reduced by designing a custom software tool. Furthermore, using such an argumentation-based approach requires minimal training and no experience. For each session, a 10-minute presentation was sufficient for the participants to be able to start using the method. The average duration of each session was 2 hours.

However, one of the main findings is that, despite our expectations, there does not seem to be a deep argument game during these assessments. This is because for each identified Risk or Attack, a suitable mitigation is found (either via a countermeasure, or by transferring or accepting the Risk). The "attacker" team can either accept the defender's argument and move on, or try to subvert the countermeasure by describing a slightly altered attack path. However, such an altered attack could, to all intents and purposes, also be viewed as a new round instead of a counter-argument (or rebuttal). So each round of the game contains at most two rounds: an attacker argument followed (optionally) by a defender counterargument.

Furthermore, the method is not affected by the presence or absence of quantitative values of likelihood and impact of Risk. This makes us optimistic that flexible Risk Assessment and/or Security Requirements elicitation tools, which can work with both quantitative and qualitative values, can be developed.

Architecture diagrams need to be more or less definite and known during the Risk Assessment as they provide crucial input. Furthermore, the scalability of Risk Assessment methods and tools increases inversely with the complexity and ambiguity of architecture. This also applies to our approach, as we noticed that if information is missing with regard to the components or technical specification of the architecture, it can drastically affect the method's utility.

Acknowledgment

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

References

[1] H. Prakken, D. Ionita, and R. Wieringa, "Risk assessment as an argumentation game." in *CLIMA*, ser. Lecture Notes in Computer Science, J. Leite, T. C. Son, P. Torroni, L. van der Torre, and S. Woltran, Eds., vol. 8143. Springer, 2013, pp. 357–373. [Online]. Available: http://dblp.uni-trier.de/db/conf/clima/clima2013.html#PrakkenIW13

[2] "Technology-supported risk estimation by predictive assessment of socio-technical security," 2012-2016. [Online]. Available: https://www.trespass-project.eu/

[3] S. E. Toulmin, *The Uses of Argument*. Cambridge University Press, July 2003.

[4] S. Toulmin, R. Rieke, and A. Janik, *An introduction to reasoning*. Macmillan, 1979.

[5] C. B. Haley, R. C. Laney, B. Nuseibeh, W. Hall, C. B. Haley, R. C. Laney, and B. Nuseibeh, "Validating security requirements using structured toulmin-style argumentation," 2005.

[6] R. M. J. D. Haley, Charles B; Laney and B. . Nuseibeh, "Arguing satisfaction of security requirements." [Online]. Available: http://oro.open.ac.uk/18878/

[7] C. Haley, R. Laney, J. Moffett, and B. Nuseibeh, "Security requirements engineering: A framework for representation and analysis," *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 133–153, Jan. 2008. [Online]. Available: http://dx.doi.org/10.1109/TSE.2007.70754

[8] V. N. L. Franqueira, T. T. Tun, Y. Yu, R. Wieringa, and B. Nuseibeh, "Risk and argument: A risk-based argumentation method for practical security." in *RE*. IEEE, 2011, pp. 239–248. [Online]. Available: http://dblp.uni-trier.de/db/conf/re/re2011.html#FranqueiraTYWN11

[9] Y. Yu, T. T. Tun, A. Tedeschi, V. N. L. Franqueira, and B. Nuseibeh, "Openargue: Supporting argumentation to evolve secure software systems." in *RE*. IEEE, pp. 351–352.

[10] H. Prakken, "An abstract framework for argumentation with structured arguments," *Argument & Computation*, vol. 1, pp. 93–124, 2010.

[11] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial Intelligence*, vol. 77, pp. 321–357, 1995.

[12] D. Ionita, *Current established risk assessment methodologies and tools*, July 2013. [Online]. Available: http://essay.utwente.nl/63830/

[13] "ENISA: Cloud Computing Security Risk Assessment," May 2009.

[14] "Top Threats to Cloud Computing V1.0," 2010. [Online]. Available: https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf

[15] A. Dutoit, R. McCall, I. Mistrik, and B. Paech, *Rationale Management in Software Engineering*. Springer, 2007.