Challenges for Risk and Security Modelling in Enterprise Architecture

Gudmund Grov, Federico Mancini, and Elsie Margrethe Staff Mestl

The Norwegian Defence Research Establishment (FFI), Kjeller, Norway
{Gudmund.Grov,Federico.Mancini}@ffi.no

Abstract. From our experience cooperating with the Norwegian Armed Forces, we outline two interconnected challenges for modelling risk and security in an enterprise architecture: (1) modelling what is protected and why it is protected with sufficient detail whilst being simple enough to facilitate analysis; and (2) establishing automated support for analysing and reasoning about the security models, something we deem crucial to exploit the full potential of an enterprise security architecture. In addition, we sketch out our approach to tackle these challenges and outline our future direction of work.

Keywords: enterprise security architecture · diagrammatic risk and security modelling · automated reasoning.

1 Introduction

One the aims of an *enterprise architecture* (EA) is to create a consistent model, or blueprint, of an enterprise's structure and organisation, from its goals and processes to its information systems. The model links different aspects, or domains, that can be considered as architectures in their own right, and which can be visualised through different *views* tailored for the specific domain and actors concerned. Security is one such domain, which we will henceforth refer to as *enterprise security architecture* (ESA), but which is often considered as one of the least developed EA domains [21]. A very important element of ESA is risk[1], and the challenges presented here are a result of ongoing work with the Norwegian Armed Forces to integrate risk and security aspects with their EA (see [8]).

There are several relevant concepts related to risk and security, which are often defined differently according to context. For this paper we use the following[2]:

Definition 1 (Threat). *Any circumstance or event with the potential to adversely impact organisational operations, organisational assets, individuals, other organisations, or the nation.*

[1]For the Norwegian defence sector security is increasingly risk-driven, and a requirement in the law of national security for classified systems.

[2]The definitions of threat and risk are adapted from NIST SP800-30 Rev. 1, while security has its origin in NIST SP800-160.

Definition 2 (Risk). *A measure of the extent to which a threat can cause adverse impact if realised (impact) and the likelihood of its occurrence.*

Definition 3 (Security). *The state of being free from unacceptable risks.*

An ESA should therefore model how unacceptable risks are dealt with at all levels of the organisation. There has been considerable work in developing theoretical frameworks for realising an ESA, such as SABSA [16], or integrating security aspects into EA frameworks [2, 13, 3, 21]. We are, however, not familiar with work focusing on more practical challenges in using and developing an ESA. During the course of our work with the Norwegian Armed Forces, two such challenges have become apparent:

1. Modelling, at a suitable level of detail, both what is protected and why it is protected, and at the same time keeping the complexity of the models at an appropriate level for analysis.
2. The lack of support for automated tools to perform analysis and reasoning about the models as their scale and complexity increases.

The main motivation for developing an ESA should be to help ensure that the modelled security measures are indeed the correct ones to handle the identified risks, and do so effectively and efficiently. This requires adequate solutions to both of our identified challenges; if not solved, the ESA may become nothing more than yet another system to statically document security. Over the next two sections we address these two challenges independently. For each, we introduce the problem with related work, and outline how we chose to approach them. Section 4 concludes the paper by addressing these challenges as a uniform problem and briefly discusses the road ahead.

2 Challenge 1: risk and security modelling in EA

Security is unique in the sense it does not provide any enterprise functionality in itself – it enables secure use of other functionalities. Without an asset to secure, or a threat to the asset, there is no point in adding security. The other peculiarity is that determining whether something is indeed secure is very hard, and depends on the context one considers. It is therefore crucial that the ESA includes both the security measures chosen for the enterprise, and the motivational aspect of security, in the form of the underlying risk, in order to provide the right context for the security. This will ease the assessment of security measures, and make their adjustment more effective in case of changes in the enterprise's risk profile. How to model this in practice is the real challenge, especially with regard to risk.

Risk and security are usually decomposed into several constituent. These are then related to one another in order to ease the systematic analysis of relevant threats and possible mitigations. Although one can find minor variations in the definitions of these factors, the underlying concept is typically that we have a threat (event), where an actor exploits one or more vulnerabilities to gain access to a valuable asset and cause a negative impact to an enterprise. Mitigations are

then put in place according to how severe the risk associated with the threat is. Many of these factors can already be modelled in an EA. Security, in particular, is relatively straightforward to model, as mitigations usually take the form of requirements, capabilities, services and processes that are already routinely modelled in an EA. Many risk factors can be modelled in EA languages such as ArchiMate [21, 3] and *Unified Architecture Framework* (UAF), however the support in *NATO Architecture Framework* (NAF) [8], which is used by the Norwegian Armed Forces, is more limited. Proposals even exists for integrating Information System Security Risk Management (ISSRM) with ArchiMate [2, 13].

The challenge is to find the right level of granularity of ESA models. On the one hand, the list of possible threats, and the level of detail in which they can be described, can quickly become unmanageable. On the other hand, a generic threat can cover multiple cases, but be almost useless in designing appropriate mitigations. Furthermore, it is not clear how risks in different domains of the EA can be placed in relation to one another to achieve traceability and consistency. A high-level business threat may correspond to several technical threats, while attacks on actual systems may have consequences on business processes that rely on them. These downwards, upwards and sideways dependencies in the EA show the need for a holistic approach to enterprise security: we must be able to trace the identified risks and threats across the EA. This contrasting need for a level of detail which conveys enough information to properly assess the enterprise-level security, but at the same time keeps complexity low enough to make analysis feasible, is a major challenge for our work.

Some assistance can come from modelling approaches to risk and security not specifically designed for ESA. *CORAS* [11] is a language developed specifically to model and analyse risk. It includes assets, vulnerabilities, threat actors, threats, unwanted incidents, risks and security controls, and supports a very granular level of detail. *Misuse cases* [17] exhibit security extensions of UML use cases to model and relate threats in the form of scenarios that could, but should not happen, together with mitigating scenarios. *Attack trees* [15] give a hierarchical depiction of attacks using AND-OR trees, where each level of the tree increases the granularity of the attack description. *Attack-defence trees* [9] augment attack trees with security controls to protect or mitigate the attacks or sub-attacks. *Bowtie diagrams* [14] enable diagrammatic separation of preventative and reactive security mechanisms in relation to the underlying threat, and have also been combined with attack trees [1]. *Assurance cases* can provide structural argumentation for the security [19], and have been studied in an EA context [22].

Figure 1 summarises the support for risk and security modelling in the approaches discussed in this section. Here, we have separated support into: *dedicated* (●), *some* (●) and *no support* (○). Some support implies that it is either partially supported or that it can be indirectly modelled through other means without dedicated support. There will naturally be some borderline cases where our classification is open for debate.

	Event	Threat	Risk	Asset	Actor	Vuln.	Impact	Mitigation
ArchiMate (AM)	●	○	◔	◔	◔	○	○	●
AM + ISSRM [2, 13]	●	●	●	●	●	●	○	●
AM extended [3]	●	●	●	◔	◔	●	○	●
UAF	◔	○	●	●	◔	○	○	●
NAF	●	○	○	◔	◔	○	○	●
CORAS [11]	◔	●	●	●	●	●	○	●
Misuse cases [17]	●	●	○	◔	●	○	○	●
Attack trees [15]	◔	●	○	○	○	○	○	○
Attack defence trees [9]	◔	●	○	○	○	○	○	●
Assurance cases [19]	○	◔	○	○	○	○	○	◔
Bowtie diagrams [14, 1]	●	●	○	◔	○	○	●	●

Fig. 1. Risk and security support. Over line: EA frameworks; under: non-EA.

2.1 Outline of our approach

In figure 1 we saw that threat is the risk factor that nearly all non-EA risk modelling approaches support. Therefore, we decided to investigate how to best model and achieve traceability of this risk component in an ESA, and leave the modelling of the other ISSRM concepts for future iterations, if at all necessary. We use NAF (version 3) as a reference EA framework, as it is what is currently used by Norwegian Armed Forces, albeit we consider both the challenges and the approach easily transferable to other frameworks, given their generic nature.

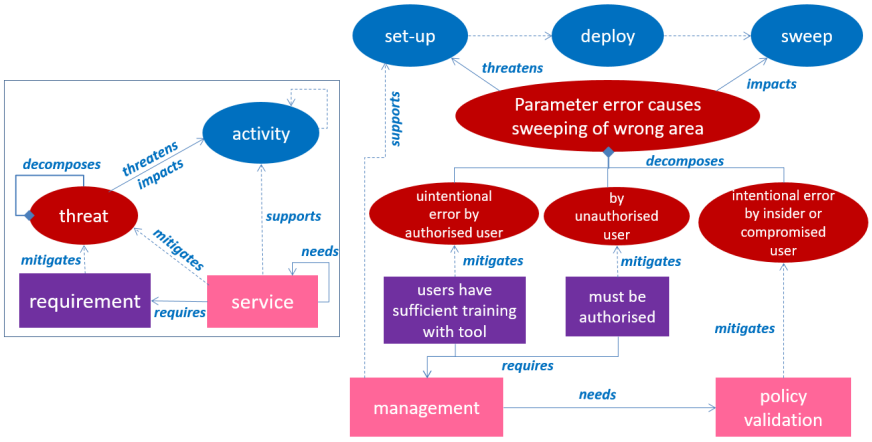

Fig. 2. Left/boxed: explanation of threat extension. Right: illustrative example.

As a first step, we propose extending NAF with a *threat* component. Figure 2 (right) shows an illustrative example using this component, while the left-hand side of the figure explains relevant components and relations. The example is based on a fragment of a mine-sweeping operation employing unmanned and

autonomous vehicles [12]. Three of the activities are shown: first the unmanned boat is configured and set-up for the mission; then it is deployed at sea where it will autonomously reach the area to sweep for mines; and finally, it will autonomously sweep for mines, based on some configuration parameters.

The threat component implicitly defines the assets to be protected and the impact of the threat, by using the *threaten* and *impact* relations, respectively. Note that both are needed as the impact may be on different parts of the model from where the assets are, and that we only show where the impact is, not its severity. In our case, the set-up activity is what needs to be protected to prevent incorrect sweeping parameters being input, and impacting the mission by causing the wrong area to be swept. We stress that the extension shown in figure 2 (left) is simplified for the example and any EA assets that need protection can be the subject of a threat. In this case it is an activity, but it can for example also be information objects, services or software.

A management service supports the configuration of the system in the set-up activity, so the threat to the activity has to be mapped to this service. In order to map threats across layers of the EA, they may need to be decomposed and refined to reflect different concerns and levels of abstraction. This may also be required as there may be sub-threats that are mitigated in different ways. To achieve this we have introduced a *decomposes* relation for threats, illustrated in the example by decomposing the overall threat into three sub-threats.

As a bridge between a threat and its mitigation we can use *requirement* components, which specify the mitigations needed to reduce the risk associated to the threat. In our example, this translates into the properties the management service should have in order to protect the set-up process from some of the identified sub-threats. This "requirement bridge" has been inspired by "safety constraints" found in the system-oriented STPA approach for safety [10] (and security [23]). Requirements are not always necessary to express the motivational aspect and achieve traceability, as in the case of the "policy validation" service, where the threat is related directly to a concrete mitigation. What one may lose in this case though, is the part of the motivational aspect used to show why a mitigation is indeed correct and sufficient for the threat. This could be useful to manage changes more efficiently at a later time. The example further shows that by decomposing the threat, we can relate specific mitigations to different aspects of the threat. This can also enable reasoning about completeness of the threat mitigation.[3]

Summary and next steps. To summarise, we have started the first iteration of our work by extending NAF with threat components which can be decomposed and traced across the EA, and used requirements as a bridge between threats and their mitigation. This bridge provides a logical abstraction of the mitigations needed to handle the threat. One interesting extension would be to use assurance cases as a formal basis for arguing that a given threat has been han-

[3] For our illustrative example the decomposition is incomplete with other sub-threats omitted for simplicity.

dled sufficiently. There are also several other ways to further extend the work. While we use a single threat element, [2] separates it into 'loss event', 'threat event' and 'vulnerability' (among others), and associates each part with different layers of the ESA. These can then be related and traced. For instance, a loss event can, via a threat event, be related to a technical vulnerability. Although such classification of different types of threats may be desirable, and will not necessary lead to larger models, we still believe a decomposition as illustrated by our example is the right way ahead as there will not always be a one-to-one mapping between threats at different parts of the ESA. Integrating other risk components beyond threats and possibly vulnerabilities, would mean integrating a complete risk analysis framework in the ESA. The question then remains as to whether this is really desirable or it should be handled by a dedicated risk framework. Our example has shown that just by incorporating threats, the size and complexity of the model is substantially increased. Further extensions may make it harder to achieve holistic reasoning about the ESA, thus reducing the effect the ESA has on improving the enterprise security. The use of a dedicated risk analysis framework, closely aligned with the ESA may be a good compromise between the conflicting need for both detail and abstraction.[4]

3 Challenge 2: automated reasoning for ESA[5]

From the previous section we may conclude that complexity is unavoidable, and accept that (purely) manual reasoning and analysis is unfeasible and automation is required. Such automation is the topic of our second challenge. As an anecdote, configurations for Amazon Web Services have become too complex to analyse manually, and automated reasoning techniques are now being used to ensure security properties of these configurations [4].

The types of question one would like to reason about in an ESA, partially mentioned in the previous section, include: Are the security controls sufficient to handle the identified risks? What are the consequences for security when changing existing processes or systems? What are the consequences for the security of the enterprise when cancelling or delaying a project? Which business processes will be impacted by given attacks or vulnerabilities?

Automated reasoning techniques require unambiguous semantics of the modelling languages, which currently do not exist for NAF, ArchiMate or UAF. Sunkle et al. [18] developed a translator for ArchiMate models that produces a representation which enables analysis of how changes made to one part of the enterprise effect other parts. It also generates a more holistic view of the technical details, which can be used for communication of the EA to a less technical audience. This illustrates two potential applications of automated reasoning. There are though reservations about such formal underpinnings to the modelling, in particular with respect to mapping the EA to other representations

[4] As a result the ESA will also act as a document management system, such as the ASCE tool by Adalard for assurance cases [https://www.adelard.com/asce].

[5] More detail about this section is available in a (Norwegian) report [7].

with conflicting semantics.[6] Such mappings are not relevant for our particular application domain, but even if they were, our belief is that multiple ways of interpreting the same model are undesirable beyond analysis. Interoperability is obviously important, but if each model has to come with an explanation of how to interpret it, then one can question why we need the model in the first place. For security, this is particularly problematic as it cuts across many aspects of an enterprise, which typically are developed by different architects (with their own semantics), and this amalgamate of different interpretations needs to be combined into a uniform and holistic judgment of the security of the enterprise. We therefore believe that a common understanding and semantics of the ESA are crucial, possibly with the parts of the models that are not required for our security analysis to be left undefined.

3.1 Outline of our approach

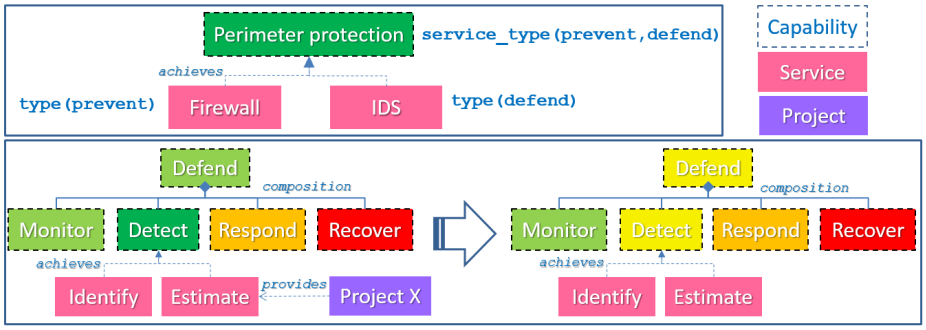

Fig. 3. Examples of security attributes and automated reasoning support.

To show feasibility for automated reasoning in an ESA context, we have developed a proof-of-concept prototype for a subset of NAF and implemented it for Sparx Enterprise Architect (EA). Several examples have been developed to illustrate a range of different usages:

Vertical coherence across architecture layers. There needs to be coherence between the architecture layers, e.g. to ensure that the enterprise operations are secured in a technical layer, and that technical security mechanisms serve a business purpose. Figure 3 (top) illustrates an example where reasoning is used to ensure that a security capability is realised at the service layer. This is augmented by additional security properties specified by the security architect, which we return to below.

Visualisation of capability strength across diagrams. *NATO CIS Security Capability Breakdown* (SCB) [6] provides a hierarchical structure for security capabilities, where each capability can be given a *strength level*,

[6] See the Open Group's ArchiMate 3 documentation [http://tiny.cc/amch5].

which indicates the maturity of the capability. This strength level can be defined recursively in terms of the strength of its sub-components, and be visualised by colouring the capability box by using a scale typically ranging from dark red (non existing) to dark green (perfect). We can use our reasoning engine to automate the colouring of capabilities, either by providing the numerical level directly or computing it from other components. Changes to one component will then automatically cause dependent capabilities to be re-coloured. Figure 3 (bottom) illustrates such colouring for a subset of SCB. Here, we can see that 'Defend' depends on other sub-capabilities and a change of colour in one of them ('Detect'), causes the recolouring of 'Defend' (rightmost diagram).

Change management. In the bottom example of figure 3 the 'Estimate' service is being delivered by a project (l.h.s. of arrow) and when this project is removed (r.h.s. of arrow) then the consequence in terms of reduced capabilities is automatically derived and visualised. In figure 3 (bottom), this is visualised by recolouring 'Detect' as a result of cancelling the project delivering the 'Estimate' service, and recolouring 'Defend' as a result of recolouring 'Detect'. This is an example of automated reasoning for change management. Changes to plans or solutions may have impacts elsewhere, e.g. cancelling, or changing the deliverables of a project may have considerable impacts on other services or capabilities of the enterprise. Our techniques can reason about direct, and indirect consequences to give an instant response to the architect, e.g. in the form of alarms or colouring as we have illustrated.

Law, regulation and policy compliance. By encoding laws, regulations and policies directly (in a formal way), compliance can automatically be verified. One simple example is the requirement for certain Evaluation Assurance Level (EAL) certification for software systems applied at given classification levels. An alarm can then be raised if the requirement is not met.

To achieve support for automated reasoning, the models are translated into a formal representation, which is analysed with the help of an external state-of-the-art automated reasoning engine [5]. Both the nodes and edges of NAF diagrams (in Sparx EA) are typed via a mechanism called UML stereotypes.[7] In addition, the nodes can be augmented with additional security properties via a mechanism we call *security attributes*[8] (SA). A SA is a piece of *formal text*, which the reasoning engine can interpret and use in the analysis. SAs are intertwined with informal natural language description of the component, and the formal and informal texts are separated by a dedicated markup.[9] We have illustrated SAs in figure 3 (top), but note that it is also the mechanism used to achieve the colouring and change management (figure 3 (bottom)). The example shows an additional security attribute, which states that the capability has to be realised

[7] For simplicity, stereotypes have been omitted in figure 3, however we have given the types of the nodes in the top corner, and some commentary for the edges.

[8] This should not be confused with the business attributes found in SABSA [16].

[9] This has been inspired by *anti-quotations* from the Isabelle proof assistant [20].

by services labelled by both 'prevent' and 'defend'. Here this is the case, but an alarm would have been raised if not.

Summary and next steps. This proof-of-concept has shown the feasibility of automated reasoning in an ESA setting, however we have only applied it to toy examples and for a limited set of problems. We have not yet attempted to reason about mitigations with regards to given threats or risks, which is an important element of our future work.

4 Towards a combined challenge

We have identified modelling and automated reasoning challenges for ESA based on our experience in the Norwegian Defence sector, and sketched out our approaches for tackling them. Whilst described as separate challenges, they are indeed interconnected: what and how is modelled will impact what we can reason about; but what we want to reason about, may change what we need to model. Both challenges are also complicated by the nature of risk and security: they are a permeating aspect of the EA and are context-dependent, so that they cannot be addressed in isolation. Finding a suitable level of abstraction and modelling it consistently to enable meaningful holistic automated reasoning is therefore crucial, since it will be impossible to analyse the ESA manually.

Security is inherently risk dependent – if there is no risk then there is no need for security. We must therefore address the reasoning challenge in a risk setting, e.g. to analyse if modelled threats and risks are sufficiently mitigated in the enterprise. This may require additional modelling elements, such as use of structured argumentation of why the threat is mitigated via assurance cases [19, 22], and possible use of bowtie diagrams to separate preventive and reactive security. The automated reasoning engine could then utilise "local arguments" about a given threat to reason holistically about the security of the entire enterprise. By solving this combined challenge, we believe the potential of ESA can be fully exploited and play a major role in securing future enterprises.

References

1. Abdo, H., Kaouk, M., Flaus, J.M., Masse, F.: A safety/security risk analysis approach of industrial control systems: A cyber bowtie–combining new version of attack tree with bowtie analysis. Computers & Security **72**, 175–195 (2018)
2. Band, I., Engelsman, W., Feltus, C., Paredes, S.G., Hietala, J., Jonkers, H., de Koning, P., Massart, S.: How to Model Enterprise Risk Management and Security with the ArchiMate Language (2017), the Open Group white paper no. W172
3. Van den Bosch, S.: Designing Secure Enterprise Architectures – A comprehensive approach: framework, method, and modelling language. Master's thesis, University of Twente (2014)
4. Cook, B.: Formal Reasoning About the Security of Amazon Web Services. In: CAV. pp. 38–47. Springer (2018)

5. De Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: TACAS. pp. 337–340. Springer (2008)
6. Gay, S.: CIS SECURITY CAPABILITY BREAKDOWN VERSION 2.00 (2017), NATO NCIA Technical Report 2017/NCB010400/13, NATO UNCLASSIFIED
7. Grov, G., Mestl, E.M.S., Mancini, F., Nordbotten, N.A.: Kan resonnering rundt sikkerhetsarkitektur automatiseres? en studie i sikkerhetsattributter og automatisk resonnering (2019), FFI-report 18-01982
8. Jørgensen, H.D., Liland, T., Skogvold, S.: Aligning TOGAF and NAF-Experiences from the Norwegian armed forces. In: PoEM. pp. 131–146. Springer (2011)
9. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Foundations of attack–defense trees. In: Formal Aspects of Security and Trust. pp. 80–95. Springer (2011)
10. Leveson, N.: Engineering a safer world: Systems thinking applied to safety. MIT press (2011)
11. Lund, M.S., Solhaug, B., Stølen, K.: Model-driven risk analysis: the CORAS approach. Springer (2010)
12. Mancini, F., Bruvoll, S., Fardal, R., Wiik, J.H., Greve, B., Olsen, L.E., Bjerketveit, B.: Information security for unmanned and autonomous vehicles – main challenges and relevant operational concepts (2019), FFI-report 19/00888 (exempt from public disclosure)
13. Mayer, N., Aubert, J., Grandry, E., Feltus, C., Goettelmann, E., Wieringa, R.: An integrated conceptual model for information system security risk management supported by enterprise architecture management. Software & Systems Modeling **18**(3), 2285–2312 (2019)
14. de Ruijter, A., Guldenmund, F.: The bowtie method: A review. Safety science **88**, 211–218 (2016)
15. Schneider, B.: Attack trees: Modelling security threats. Dr. Dobb's Journal of Software Tools **24**(12), 21–29 (1999)
16. Sherwood, N.A.: Enterprise security architecture: a business-driven approach. CRC Press (2005)
17. Sindre, G., Opdahl, A.L.: Eliciting security requirements with misuse cases. Requirements engineering **10**(1), 34–44 (2005)
18. Sunkle, S., Kulkarni, V., Roychoudhury, S.: Analyzing enterprise models using enterprise architecture-based ontology. In: MODELS. pp. 622–638. Springer (2013)
19. Weinstock, C.B., Lipson, H.F., Goodenough, J.B.: Arguing Security – Creating Security Assurance Cases (2007), white paper by the Software Engineering Institute (Carnegie Mellom University)
20. Wenzel, M., Chaieb, A.: SML with antiquotations embedded into Isabelle/Isar. In: Workshop on Programming Languages for Mechanized Mathematics (2007)
21. Wierda, G.: Mastering ArchiMate Edition III: A serious introduction to the ArchiMate enterprise architecture modeling language. R&A (2017)
22. Yamamoto, S., Kobayashi, N.: Mobile Security Assurance through ArchiMate. IT CoNvergence PRActice (INPRA) **4**(3), 1–8 (2016)
23. Young, W., Leveson, N.G.: An integrated approach to safety and security based on systems theory. Communication of the ACM **57**(2), 31–35 (2014)