Contextualizing of Architectural Security Patterns as a Knowledge Management Challenge

Andrei Brazhuk,
Yanka Kupala State
University of Grodno
Grodno, Belarus

Evgeny Olizarovich
Yanka Kupala State
University of Grodno
Grodno, Belarus

Abstract. **Security-by-design as adoption of security solutions for a system design is in focus of this work. This field is treated as requiring expert knowledge and heavy for automation. A perspective way to improve exiting security design methodologies is the use of security patterns as a mechanism of collecting secure design artifacts. To apply security patterns as a part of automation of secure design, it requires well-formed collections of security patterns and innovative method to support the design decisions.**

This work considers a contextualizing challenge as a way to define the necessity of a security pattern in a given case. Understanding of context includes two main questions: "Is the security pattern suitable for a system design?" and "Does the security pattern affect a particular security challenge?".

We approach a direct architectural contextualizing as a basic mechanism of automatic mapping of security artifacts (threats, security solutions) to components of a computer system during early design stages (requirements, design). Also, this work describes two use cases of the architectural contextualizing based on an ontological cloud threat pattern catalog: the use of a query language for finding relevant security patterns and analysis of graphical system representations based on an ontology driven threat modeling.

This work uses a strict ontological approach, implemented with Web Ontology Language (OWL) and automatic reasoning procedures.

Keywords**: security pattern, ontology, contextualizing, threat modeling, OWL**

I. INTRODUCTION

Security methodologies solve various challenges of secure development by improving security attributes of computer systems [1]. They describe security as a set of processes (threat modeling, risk management, secure design, etc.) and operate different artifacts (threats, controls, mitigations, metrics etc.) of conceptual security models.

Security-by-design is in focus of this work, i.e., adoption of security solutions to a particular design. This is commonly considered as an informal field, required expert knowledge, and it is most challenged from the automation point of view. They need well-formed collections of artifacts, also methods and algorithms to take right decisions.

Security patterns are known as a way of representation of various security artifacts (especially holding architectural decisions in some form) and reusing them. They are important in improvement efficiency of security methodologies: making the process iterative, and integration of the threat modeling and the secure design subprocesses. The common approach is to use different artifacts at early lifecycle stages: use cases and abstract security patterns at the requirements stage, and threat taxonomy and concrete security patterns at the design stage. Note, dealing with threats can also be possible with a special kind of security patterns, called threat patterns.

A security pattern is considered as a class, and applying it to a design is called 'instantiation'. Having a description (texts, diagrams, artifacts) of a system with flaws and vulnerabilities, it requires to correct items of the description from security perspective, injecting adequate security patterns. Instantiation as a complicated process is out of scope of this work, as well as its possible supplementary processes like integration and verification [2].

All the things that define the necessity of the pattern in the design we call 'contextualizing' in this work. Understanding of context includes two main questions: "Is the security pattern suitable for the system design?" and "Does the security pattern affect a particular security challenge?".

Contextualizing can be done manually (semi-automatically) or automatically.

Manual (semi-automatic) use case is like: an architect works with a system architecture, depicted as text or as a graphical notation, like UML. To help the architect to choose a pattern (or an ordered set of patterns) out of several hundred existing, a security pattern catalog can be used. The catalog can contain different labels and some sort of a query language can be used to find relevant patterns.

Automatic use case is based on a formal scenario that describes a computer system in general, the

Pattern Recognition and Information Processing (PRIP'2021) : Proceedings of the 15th International Conference, 21–24 Sept. 2021, Minsk, Belarus. – Minsk : UIIP NASB, 2021. – 246 p. – ISBN 978-985-7198-07-8.

scenario is used to build the system automatically by an orchestration system software. To analyze security aspects of such applications it could be useful apply automation, which allows to correct the deployment scenario by automatic implementation of a relevant security pattern, or automatically check dependent components for meeting of requirements of current application's SLA. Advanced knowledge management techniques and the Artificial Intelligence (AI) technologies should be used to implement the automatic use case.

This work considers essential items of contextualizing, and contributes by a description of a direct architectural contextualizing as a basic mechanism of automatic mapping of security artifacts (threats, security solutions) to components of a computer system during early design stages. It is a part of an ontological schema of security patterns, based on a strict ontological approach and implemented with Web Ontology Language (OWL) and automatic reasoning procedures.

Also, this work describes two use cases of the architectural contextualizing based on an ontological cloud threat pattern catalog. The first use case shows the usage of query language for finding relevant threat patterns. The second one depicts the analysis of a simple graphical system representation with data flow diagrams (DFD) based on an ontology driven threat modeling.

The rest of this work is structured as follows. Section II shows related works in the threat modeling and security pattern fields. Section III describes the direct architectural contextualizing. Section IV illustrates its use cases. And a summary of the results and future research is discussed in Conclusions.

II. Related work

Traditionally, threat modeling is considered as a semi-automatic process, happening at early stages (requirements, design) of system lifecycle. A challenge is to increase its automation part, because this gives opportunities to enable automatic contextualizing. Also, moving of threat modeling to run-time with approaches like reflective threat modeling [3] is in research focus now, these will require advanced knowledge management techniques. That is another reason for importance of the contextualizing challenge in automatic threat modeling.

Existing efforts of the contextualizing of common security knowledge are primary based on rule-based languages, graphs, domain specific languages (DSL), logics, and ontologies.

Work [4] has proposed to use a catalog of security patterns to automatically detect vulnerabilities in a software architecture. Their patterns have been defined with a graph language, and context should be determined by appropriate query rules. Several works [5, 6] are known as a continuation of use of rule-based languages to automate threat modeling.

Works [7, 8] have described efforts to apply a meta language for creation of domain specific languages to depict security challenges and its enhancement for cyber-attack scenarios, in particular with probability distributions. They have used attack graphs as implementation.

Work [9] has applied a bit of logic-based approach with Prolog based rules to define context.

Also, there are works that have described an ontological approach of conceptual structures of security frameworks [10, 11], in particular focused on the threat modeling [12, 13].

From our perspective, ontologies based on strict formalization (e.g., OWL and automatic reasoning) most meet the automation challenges of the threat modeling. Work [14] has described a framework of ontology driven threat modeling that potentially supports contextualizing based on security labels (CIA, STRIDE) and architecture.

Despite collecting the security patterns for decade [15, 16], several challenges of their use exist, like lack of approaches to recognize necessity of security patterns for a computer system design. There are diverse researches aimed at formalizing security patterns and creation their catalogs, and most of patterns are represented by the UML diagrams with text descriptions in the POSA format (at least it requires to fill the context, problem and solution fields). So, it can be considered two directions of research, related to contextualizing: both transformation of the diagrams and ordering of the textual meta information into knowledge-like formats (graphs, ontologies, etc.).

Work [17] has approached a security pattern classification, based on transformation of the UML descriptions to the Attack Deference Trees (ADT). Work [18] has described a security pattern detection framework in order to put them in a security pattern graph database.

Work [19] has proposed a modeling language to define security patterns based on metamodeling techniques. And work [20] has described a conceptual approach of interconnecting various pattern languages.

There are several researches, aimed to classify metadata of security patterns [21], up to creation of ontology-driven tools [22].

Pattern Recognition and Information Processing (PRIP'2021) : Proceedings of the 15th International Conference, 21–24 Sept. 2021, Minsk, Belarus. – Minsk : UIIP NASB, 2021. – 246 p. – ISBN 978-985-7198-07-8.

And several catalogs [23, 24] and domain specific models [25, 26] of security patterns have been developed.

However, it can be argued that lack of efforts exists in order to unite the threat modeling technologies and security pattern approach, also apply different automatic techniques and methods into the secure development process. And resolving of the contextualizing challenge can be a step forward in this field.

III. Direct architectural contextualizing

Direct architectural contextualizing is a part of the ontological schema of security patterns [22]; the schema also represents common features of patterns, like idea, author, type, their hierarchy and relationship, and a set of characteristics used by the scientific community.

The ontological schema, implemented with OWL, allows creation of well-formed catalogs of security patterns. From architectural point of view a catalog includes two items.

First item is a metamodel of given domain with a hierarchy of typical components, implemented by the class-subclass relationship (for example, our cloud specific catalog, described below, includes components like Cloud Infrastructure, Cloud Application, Remote User; and Cloud Applications can be divided to Virtual Machines, PaaS Applications, and SaaS Applications).

Second item includes strict descriptions of security patterns with the security and context labels.

Note, architectural context is independent from pattern type, so it can be possible to apply the same approach to the security patterns, misuse patterns and threat patterns.

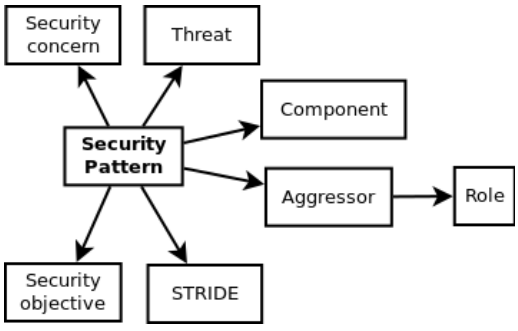

Fig. 1. Structure of direct architectural context

For example, it is considered interaction between remote user and cloud application (see Fig. 2) and it requires to put into context a security pattern that affects some destructive activity from remote user. In this case remote user is treated as an aggressor, and cloud application is an affected component. Additionally, it can be possible to apply role of the aggressor (client) in order to keep information about direction of the network connection.

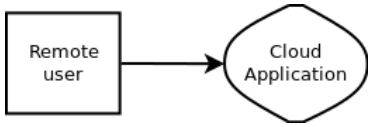

Fig. 2. Context example

And, in addition, it can be possible to apply security characteristics of the pattern. In general, this includes two points of view (Fig. 1): view of a security expert, represented by the threat concept, and view of an architect, represented by the security concern concept.

Depending of modeling goals different approaches can be used to define security labels. Threat taxonomy can be built from the CAPEC enumeration, what gives capability to map them with the ATT&CK, CWE, CVE enumerations [27, 28]. or use one of several original approaches [29, 30].

Security concerns are considered as security features that a security pattern holds in terms of software requirements. It can be useful to consider security control families from the NIST SP 800-53 publication as security concerns. In theory this enables mapping of security pattern catalogs with different security control catalogs [31, 32].

Also, the common security labels are used via the security objectives (CIA) and STRIDE concepts (Fig. 1).

Having a set of architectural (component, aggressor, role) and security (concern, threat) labels, it can be possible to make requests to a catalog in order to find relevant patterns. Also, strict contextualizing enables automatic procedures of comparing design templates, created from the architectural labels, and items of system description, made in some graphical or text notation, in order to define applicability of a security pattern.

IV. Cloud computing use case

Currently, creation of a security pattern catalog includes two stages. Firstly, an ontology, based on the schema [22], should be created to describe concepts and instances of a specific computing environment. Then a JSON-schema file from the ontology can be generated by a simple tool. Secondly, pattern descriptions can be created as JSON files with a JSON-schema based editor. The simple tool allows generating of an ontology of security patterns from the pieces of JSON.

We have been developing the Academic Cloud Computing Threat Patterns (ACCTP) catalog [33] (https://nets4geeks.github.io/acctp/) to research

Pattern Recognition and Information Processing (PRIP'2021) : Proceedings of the 15th International Conference, 21–24 Sept. 2021, Minsk, Belarus. – Minsk : UIIP NASB, 2021. – 246 p. – ISBN 978-985-7198-07-8.

feasibility of the ontological approach of management of security patterns. Threat pattern catalog is a kind of security pattern catalog, intended to collect architectural threats of a particular domain and used for analysis of security use cases and creation of a threat taxonomy. Our implementation of ACCTP has also included references to the common cloud security solutions to make it more close to the real security challenges.

A. *Finding relevant threat patterns*

To illustrate this use case, we use the Protege ontology editor and the DL query language. For example, to show all the threats the Cloud Application concept can be affected, you can use a DL sentence like:

hasAffectedComponent some CloudApplication

To apply the STRIDE filter to the previous query, you can use request (its results are shown in Fig. 3):

hasAffectedComponent some CloudApplication and hasSTRIDE value STRIDE_Denial_of_Service

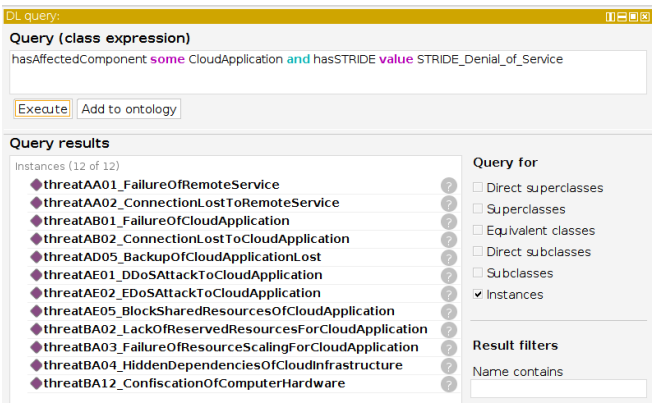

Fig. 3. Example of DL query

B. *Analysis of graphical system description*

Several graphical notations, like data flow diagrams (DFD) and process flows exist that be used to apply flow-based threat modeling.

In order to enable an ontological approach of DFD analysis, we have converted the ACCTP ontology to an appropriate ontological domain specific threat model [33], we have created a console modeling tool, and adopted third-party GUI threat modeling tool (OWASP Threat Dragon). The ontologies of a base threat model, the ACCTP domain specific threat model, and semantic interpretation of a diagram should be processed by automatic reasoning procedures to get a list of threats for a given system description.

Fig. 4 shows threats that touch remote cloud users, interacting with a cloud application. Threats are taken the catalog by the automatic reasoning procedures, and this 'automatic' decision is based on a sort of an ontological flow template that catches such kind of interactions.

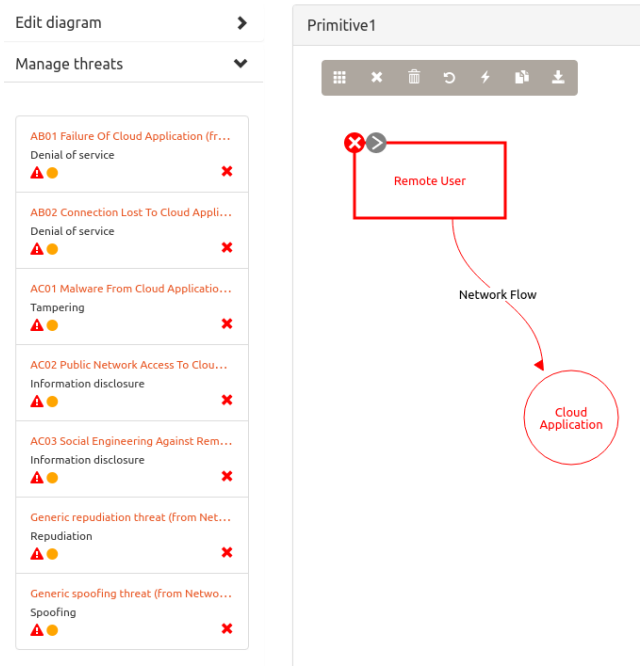

Fig. 4. Results of automatic threat modeling

V. Conclusion

This work contributes the direct architectural contextualizing as a basic mechanism of automatic mapping of security artifacts (threats, security solutions) to components of a computer system during the requirements and design stages. It can be possible strictly define a context of a security (threat) pattern with proposed properties. However, direct contextualizing is a naive approach that can be used as a proof of concept and for restricted use cases of semi-automatic secure design. To extend it, advanced knowledge management and decision support (AI-like) technologies are needed. Additionally, the challenge of automatic instantiation should be considered because strict contextualizing depends on such a challenge.

Also, this work describes the use cases of the contextualizing based on an ontological cloud threat pattern catalog (use of a query language and ontology driven threat modeling of data flow diagrams). Note, industry adoption of the use cases requires creation of software tools and modules, and redesign of existing security methodologies in order to integrate threat modeling and design itself (DFDs are not considering as a good design approach).

Mentioned above challenges form possible directions of future research.

Pattern Recognition and Information Processing (PRIP'2021) : Proceedings of the 15th International Conference, 21–24 Sept. 2021, Minsk, Belarus. – Minsk : UIIP NASB, 2021. – 246 p. – ISBN 978-985-7198-07-8.

References

[1] A.V. Uzunov, E.B. Fernandez, K. Falkner, "ASE: A comprehensive pattern-driven security methodology for distributed systems." Computer Standards&Interfaces 41, 2015, pp. 112-137.

[2] T. Peng, S. Wang, J. Geng, Q. Wang, "Verification of the Instantiation and Integration of Security Patterns." Journal of Web Engineering , 2020, pp. 521-556.

[3] D. Van Landuyt, L. Pasquale, L. Sion, "Threat models at run time: the case for reflective and adaptive threat management (NIER track)," SEAMS'21: Proceedings of the 16th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 2021.

[4] B.J. Berger, K. Sohr, R. Koschke, "Automatically extracting threats from extended data flow diagrams," International Symposium on Engineering Secure Software and Systems. Springer, Cham, 2016.

[5] S. Peldszus, "Model-driven Development of Evolving Secure Software Systems," Software Engineering (Workshops), 2020.

[6] K. Tuma, at. al., "Automating the early detection of security design flaws," Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, 2020, pp. 332-342.

[7] S. Katsikeas, et al., "An attack simulation language for the IT domain," International Workshop on Graphical Models for Security, Springer, Cham, 2020, pp. 67-86.

[8] W. Xiong, S. Hacks, R. Lagerström, "A Method for Assigning Probability Distributions in Attack Simulation Languages," Complex Systems Informatics and Modeling Quarterly, 2021, vol. 26, pp. 55-77.

[9] S. Hahner, et al., "Modeling Data Flow Constraints for Design-Time Confidentiality Analyses," 2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C), 2021, pp. 15-21.

[10] R.Y. Venkata, et al., "A Domain-agnostic Framework for Secure Design and Validation of CPS Systems," International Journal on Advances in Security, Vol. 13, Num. 3 & 4, 2020.

[11] V. Vassilev, et al., "Intelligence graphs for threat intelligence and security policy validation of cyber systems," Proceedings of International Conference on Artificial Intelligence and Applications, Springer, Singapore, 2021, pp. 125-139.

[12] M. Välja, et al., "Automating threat modeling using an ontology framework," Cybersecurity, vol. 3, no. 19, 2020.

[13] A. Shaked, Y. Reich, "Model-based Threat and Risk Assessment for Systems Design," 7th International Conference on Information Systems Security and Privacy (ICISSP 2021), 2021, pp 331-338.

[14] A. Brazhuk, "Security patterns based approach to automatically select mitigations in ontology-driven threat modelling," Open Semantic Technologies for Intelligent Systems (OSTIS), 2020, pp. 267-272

[15] H. Washizaki, et al., "Systematic Literature Review of Security Pattern Research," Information, 2021, vol. 12, no. 1.

[16] A.J. Jafari, A. Rasoolzadegan, "Security patterns: A systematic mapping study," Journal of Computer Languages, 2020, vol. 56.

[17] S. Salva, L. Regainia, "An Advanced Approach for Choosing Security Patterns and Checking their Implementation," arXiv preprint, arXiv:2007.03275, 2020.

[18] A.K. Alvi, M.A. Zulkernine, "A security pattern detection framework for building more secure software," Journal of Systems and Software, 2021, vol. 171.

[19] B. Hamid, S. Gürgens, A. Fuchs, "Security patterns modeling and formalization for pattern-based development of secure software systems," Innovations Syst. Softw. Eng. vol. 12, no. 2, 2016, pp. 109–140.

[20] M. Weigold, et al., "Pattern Views: Concept and Tooling for Interconnected Pattern Languages," Symposium and Summer School on Service-Oriented Computing, Springer, Cham, 2020, pp. 86-103.

[21] M. VanHilst, et al., "A multi-dimensional classification for users of security patterns,". J. Res. Pract. Inf. Technol. vol. 41, no. 2, 2009, pp. 87–97.

[22] A. Brazhuk, E. Olizarovich, "Format and Usage Model of Security Patterns in Ontology-Driven Threat Modelling," Russian Conference on Artificial Intelligence, Springer, Cham, 2020, pp. 382-392.

[23] N. Marko, A. Vasenev, C. Striecks, "Collecting and Classifying Security and Privacy Design Patterns for Connected Vehicles: SECREDAS Approach," International Conference on Computer Safety, Reliability, and Security, Springer, Cham, 2020. pp. 36-53.

[24] M. Papoutsakis, et al., "Towards a Collection of Security and Privacy Patterns," Applied Sciences, 2021, vol. 11, no. 4.

[25] R. Saemaldahr, et al., "Reference Architectures for the IoT: A Survey," Innovative Systems for Intelligent Health Informatics. IRICT, Lecture Notes on Data Engineering and Communications Technologies, vol 72. Springer, Cham, 2021.

[26] C. Silva, et al., "Contract-based design patterns: a design by contract approach to specify security patterns," Proceedings of the 15th International Conference on Availability, Reliability and Security, 2020, pp. 1-9.

[27] K. Kanakogi, et al., "Tracing CAPEC Attack Patterns from CVE Vulnerability Information using Natural Language Processing Technique," Proceedings of the 54th Hawaii International Conference on System Sciences, 2021.

[28] C.B. ŞAHİN, "The Role of Vulnerable Software Metrics on Software Maintainability Prediction," Avrupa Bilim ve Teknoloji Dergisi, 2021, no. 23. pp. 686-696.

[29] A.V. Uzunov, E.B. Fernandez, "An extensible pattern-based library and taxonomy of security threats for distributed systems," Computer Standards & Interfaces, 2014, vol. 36, no. 4, pp. 734-747.

[30] A. Massel, D. Gaskova, "Identification of Critical Objects in Reliance on Cyber Threats in the Energy Sector," Acta Polytechnica Hungarica, 2020, vol. 17, no. 8.

[31] K.P. Joshi, L. Elluri, A. Nagar, "An Integrated Knowledge Graph to Automate Cloud Data Compliance," IEEE Access, 2020, vol. 8.

[32] V.I. Vasilyev, A.M. Vulfin, L.R. Chernyakhovskaya, "Cybersecurity Risk Analysis of Industrial Automation Systems on the Basis of Cognitive Modeling Technology," Digital Forensic Science, IntechOpen, 2019.

[33] A. Brazhuk, "Threat modeling of cloud systems with ontological security pattern catalog," International Journal of Open Information Technologies, vol. 9, no. 5, 2021, pp. 36-41.

Pattern Recognition and Information Processing (PRIP'2021) : Proceedings of the 15th International Conference, 21–24 Sept. 2021, Minsk, Belarus. – Minsk : UIIP NASB, 2021. – 246 p. – ISBN 978-985-7198-07-8.