From Secure Business Process Modeling to Design-Level Security Verification (Artifact Paper)

Qusai Ramadan*, Mattia Salnitri†, Daniel Strüber*, Jan Jürjens*‡ and Paolo Giorgini†
*University of Koblenz-Landau, Koblenz, Germany
Email: {qramadan, strueber, juerjens}@uni-koblenz.de
†University of Trento, Trento, Italy
Email: {mattia.salnitri, paolo.giorgini}@unitn.it
‡ Fraunhofer-Institute for Software and Systems Engineering ISST, Dortmund, Germany

***Abstract*—We present the artifact submission for our paper of the same name, to be presented at the MoDELS conference 2017 in Austin, TX. Our submission includes the model transformation from SecBPMN2 to UMLsec models as well as four example SecBPMN2 models from the Air Traffic Management System case study. We explain the process of using the transformation, and the verification of the generated UMLsec models using the CARiSMA tool.**

I. INTRODUCTION

In this paper, we present the artifact submission for our paper of the same name [1], to be presented at the MoDELS conference 2017 in Austin, TX. Our submission includes the model transformation from SecBPMN2 [2] to UMLsec [3] models as well as four example SecBPMN2 models from the Air Traffic Management System case study. The models created for the case study are provided online at `https://figshare.com/account/projects/23464/articles/5223928` We explain the process of using the transformation, and the verification of the generated UMLsec models using the CARiSMA [4] tool.

In Fig.1, we show an artifact-centric representation of the process for applying our framework, including two automated tasks. The first one is an automated model transformation from SecBPMN2 models to corresponding UMLsec structural diagrams (i.e., deployment and class diagrams), using the model transformation language Henshin and its associated toolset [5], [6]. This task is implemented using a set of transformation rules (.henshin files) and some Java code for rules orchestration. The rules are defined graphically and applied to the input models (i.e., SecBPMN models) via an interpreter engine provided by Henshin. The output of this task is a UMLsec model, and a trace model. The trace model links the SecBPMN2 and UMLsec models. Using the trace models, one can check whether a UMLsec security stereotype is in place for each security annotation specified in the SecBPMN2 model.

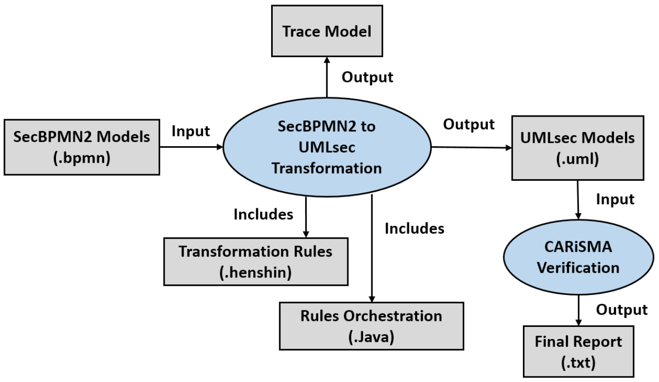

Fig. 1: Process with involved tasks and artifacts

In the second task, we use CARiSMA to automatically verify the generated UML models against UMLsec policies. The output of this process is a text file that summarizes the results of the verification process. More details about how to use our approach are provided in Sect. II

REFERENCES

[1] Q. Ramadan, M. Salnitri, D. Strüber, J. Jürjens, and P. Giorgini, "From Secure Business Process Modeling to Design-Level Security Verification," (Accepted).

[2] M. Salnitri, F. Dalpiaz, and P. Giorgini, "Designing secure business processes with SecBPMN," *Software & Systems Modeling*, pp. 1–21, 2016.

[3] J. Jürjens, *Secure systems development with UML*. Springer Science & Business Media, 2005.

[4] "CARiSMA," `https://rgse.uni-koblenz.de/carisma/`.

[5] T. Arendt, E. Biermann, S. Jurack, C. Krause, and G. Taentzer, "Henshin: advanced concepts and tools for in-place EMF model transformations," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2010, pp. 121–135.

[6] D. Strüber, K. Born, K. D. Gill, R. Groner, T. Kehrer, M. Ohrndorf, and M. Tichy, "Henshin: A usability-focused framework for emf model transformation development," in *International Conference on Graph Transformations*, 2017, pp. 125–141.

[7] "ReadMe: Description of SecBPMN2 to UMLsec Transformation," `https://github.com/grammarware/models17/blob/master/ramadan/README.md`.

II. USAGE

Prerequisite. To be able to execute our project, we recommend using Eclipse Neon, Modeling Tools distribution from https://www.eclipse.org/downloads/packages/eclipse-modeling-tools/neonr, with an installed nightly build of Henshin and CARiSMA. These softwares plug-ins can be installed on your Eclipse (Help →Install New Software...) from the the follwing update sites: for CARiSMA use http://carisma.umlsec.de/updatesite, while for Henshin one can use http://download.eclipse.org/modeling/emft/henshin/updates/release. From the CARiSMA update site, please only install the main features (BPMN2 and UML2 support).

Performing the transformation. To execute the transformation from SecBPMN2 to UMLsec models, please mind the following instructions. More details are available in the *ReadMe* file https://github.com/grammarware/models17/blob/master/ramadan/README.md.

- Download and import our project package *"myexample"* to your local Eclipse workspace https://figshare.com/account/projects/23464/articles/5223919, via File → Import → From Archive File...
- Right click on the main class src/my.example/BpmnToUml.java → RUN As *JUnit Plug-in Test* to perform the transformation. By default, our transformation takes the *example1.bpmn* file as input. To change the input file, first copy the name of one of the BPMN files that are provided in the *myexample* → *src* → *my.example* directory. Second, find line 91 (**public static final String EXAMPLE = "example1.bpmn";**) in the *BpmnToUml.java* file (line no. xyz)and replace the file name *"example1"* with the name of the selected BPMN file.
- After executing the *BpmnToUml.java*, you should see console output informing you about the generation process. The process could take a few minutes, and there might be some warnings/error messages related to the underlying plug-ins. As these do not concern us, we can ignore them. The process is finished when the following line is printed to the console: Saved result in "example1-generated-result.uml".
- The results of the transformation process (.uml file) will be stored to the *myexample/src/my.example/* directory. The name of the UML file is *example1-generated-result.uml*. You may have to refresh your Package Explorer (press F5) to see it.

Performing the verification. In this step, we use CARiSMA checks to verify the generated UML models against UMLsec security policies.

- Right click on the *myexample* project → New → Other → CARiSMA → Analysis → Next in the dialog, select the file that is generated from the last step (i.e., example1-generated-result.uml) and then click finish.
- From the dialog, click on *add checks to the list* icon → select the check that you want to perform (e.g., *secure links UMLsec check* and *secure dependency UMLsec check*) then click run. For «abac» policy, you have to select both *RABACsec: Create transformation input* and *RABACsec: Use transformation input* checks. The former allows you to select the role that you want to verify his accessibility to the system operations, while the later return the set of operations that the selected role has an access to them. More details about the execution of CARiSMA checks are provided in the ReadMe file [7]. Other information can also be found in the user manual of CARiSMA. After installing CARiSMA, the manual is available under: Help → Help Contents → CARiSMA.
- The result of the verification is provided in the *Analysis Results view*. One can also right click on the result and select *create a report for selected analysis*. The report will be stored to the *myexample/src/my.example/* directory.