

# Artifact - Enhancing Threat Model Validation: A White-Box Approach based on Statistical Model Checking and Process Mining

Roberto Casaluce, Andrea Burattin, Francesca Chiaromonte,  
Alberto Lluch Lafuente, Andrea Vandin

March 12, 2024

## 1 Short abstract to the documentation

This present research enhances the validation of threat models by leveraging the combined capabilities of Statistical Model Checking (SMC) and Process Mining (PM). Our methodology SMC plays a crucial role in accurately estimating system properties, while PM is utilized to analyze execution logs and uncover essential process patterns. Indeed, the integration of SMC and PM provides a synergistic advantage, as it yields invaluable insights into the system dynamics, thereby facilitating the efficient identification of potential issues and it provides an automatic remedy for deadlocks discovered during the analysis of threat models studied using RisQFLan, a domain-specific instantiation of QFLan applied to the risk modeling and analysis domain.

## 2 Introduction

This document offers instructions for utilizing the scripts employed in the experiments conducted as part of the research paper titled "Enhancing Threat Model Validation: A White-Box Approach with Statistical Model Checking and Process Mining." These scripts are crafted to perform a diff analysis between procedural models outlined in the DOT files and event logs derived from simulations of the RisQFLan models. Additionally, the second script also aids the modeler in resolving deadlocks identified within the model by automatically generating a refined model that fixes those deadlocks.

The following subsections will explain how to install the requirements and how to run each of the different types of experiments.

## 3 Prerequisites and installation

- Ensure `pip` is installed.

```

ArtifactDAMOCLES.zip/
├── DiffModel_RefinedModel/
└── ModelsRisQFLan/

```

Figure 1: Folder zip file project.

1. Download the project.<sup>1</sup>
2. Create a fresh virtual environment (optional but recommended).
3. Navigate to the project directory and install the required dependencies using `pip`:

```
pip install -r requirements.txt
```

- Python version  $\geq 3.9$  (Tested with Python version 3.10.6 on Ubuntu 22.04.2 LTS).

Note: On Windows, additional steps are required to install the `pygraphviz` package. Refer to the official documentation for more information: <https://pygraphviz.github.io/documentation/stable/install.html#advanced>. For Macs see <https://github.com/pygraphviz/pygraphviz/issues/398>

## 4 Structure of the zip file

The zip file *ArtifactDAMOCLES.zip* contains two folders. The first one comprises the differential model scripts, along with another script. This additional script not only generates a differential model but also produces a refinement model that addresses deadlocks found in the simulated model, i.e., *DiffModel\_RefinedModel*, and the second folder includes the original models to run the experiments on RisQFLan from scratch, i.e., *ModelsRisQFLan* - see Figure 1.

### 4.1 ArtifactDAMOCLES

#### 4.1.1 Project Structure

Figure 2 illustrates the project directory. The `data` directory contains several folders, some used to store files for analysis, and others for saving results. In the `csv`, `dot`, and `bbt` folders, event logs files, dot files, and the RisQFLan file are stored, respectively. The `results.DIFF` folder will store the diff model in both PDF and DOT formats. Meanwhile, the `refined.BBT` folder is used to store the preview of the new refined attacker behavior (AB) graph and the refined model, with the latter saved in both TXT and BBT formats.

<sup>1</sup>Check also here [https://github.com/rcasaluze/diff\\_model\\_refinement](https://github.com/rcasaluze/diff_model_refinement) for updates.

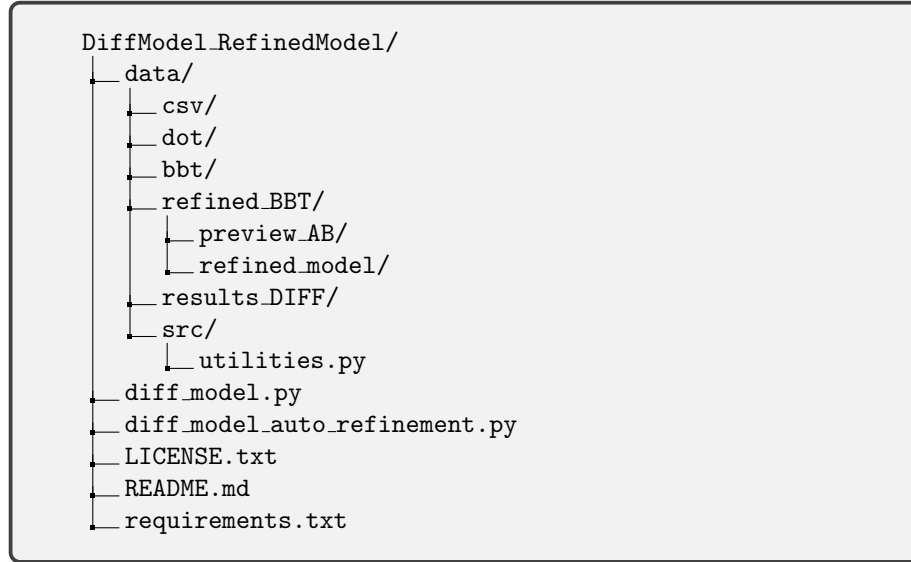


Figure 2: Folder structure of the project.

Figure 3 illustrates the core aspects of our method. The primary input for the process consists of event logs mined using PM techniques. The PM model obtained is then converted into a graph that represents the procedural aspect of RisQFLan. Then, this new model is compared with the second input, i.e., the original model designed by the modeler, and the final result is the diff model highlighting the existing differences between the two models. This method is employed for both scripts, i.e., `diff_model.py` and `diff_model_auto_refinement.py`. only the latter can provide a refinement model when deadlocks are found in the model.

#### 4.1.2 Options for `diff_model.py`

The `diff_model.py` script performs a diff analysis between the graphical representation of the procedural part of the RisQFLan model and the mined PM model.

- `-h, --help`: Show the help message and exit.
- `-i INPUTCSV, --inputcsv INPUTCSV`: Input CSV file - simulated event logs saved from RisQFLan.
- `-idot INPUTDOT, --inputdot INPUTDOT`: Input DOT file - the graphical representation of the procedural part of the RisQFLan model.

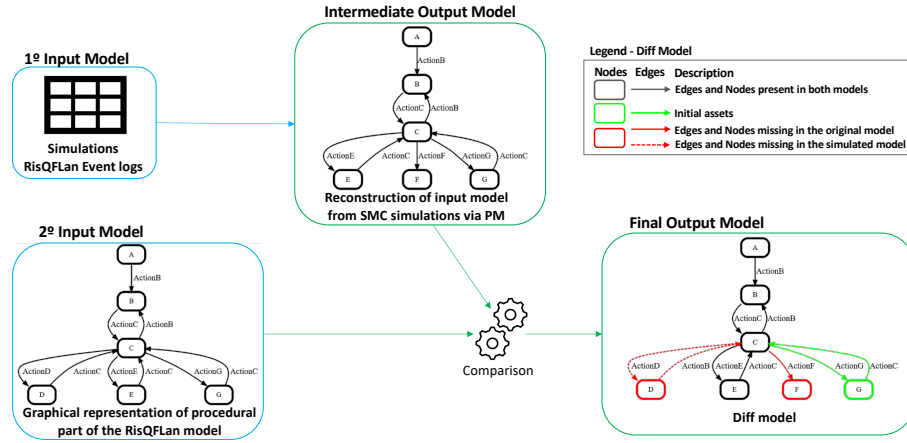


Figure 3: Example of the input and output produced by the technique presented in this paper. The 1<sup>st</sup> Input model is simulated with SMC techniques and corresponding traces are used to synthesize a new model - *Intermediate output model* - which is then compared to the original one - 2<sup>nd</sup> Input model - and an easy-to-read output is returned to the modeler emphasizing the differences - *Final output model*.

- `-o OUTPUT, --output OUTPUT`: Output NAME - diff model between the graphical representation of the procedural part of the RisQFLan model and the mined PM model.

Example:

```
python diff_model.py -i [CSV_file].csv -idot [DOT_file].dot
-o [OUTPUT_file].pdf
```

The diff model is saved in the folder **results\_DIFF**. During the analysis to create the diff model, the user will be prompted to enter, if any, the initial assets that the attacker had prior to the attack. This is important because if there are nodes already considered as initial assets and they are not singled out from the other nodes, the diff model may mistakenly assign to them the edges and nodes of the initial assets as missing in the simulated logs. Consequently, they will be colored in red.

To conduct the experiments and obtain only the diff model included in the paper:

```
python diff_model.py -i log_RobBank_original.csv -idot
RobBankAttacker.dot -o diff_model_w_diff_script.pdf
```

After executing the aforementioned commands, the user will be prompted to input, if any, the nodes already acquired by the attacker. These nodes represent the initial assets of the attacker, and, in this case, the user will need to type the node **FindCode1** to replicate the same diff model as described in the paper.

#### 4.1.3 Options for `diff_model_auto_refinement.py`

The `diff_model_auto_refinement.py` script also conducts a diff analysis between the graphical representation of the procedural part of the RisQFLan model and the mined PM model. Additionally, if the simulated model incorporates deadlocks, the script will automatically generate a new refined model and save it as a RisQFLan model. Figure 4 depicts the steps involved in automatically refining the model. Firstly, this script recycles the mined model - referred to as the 'Intermediate Output Model' in Figure 3 - used to create the diff model. If the simulated model has at least one deadlock, the script will extract all the components included in the BBT file, located in the folder `BBT`. Then, it will create new transitions in the original transition system to resolve the deadlocks found, and the script will populate an empty RisQFLan template with all the information from the original model and the new transition system which is saved in the `refined_model`. Furthermore, the script will save a preview of the new attacker behavior model in the folder `preview_AB`, where the new transitions are colored in orange to distinguish them from the other original transitions, which are colored in blue. To use the new refined model, the user must copy the text in the model and paste it into the RisQFLan software in an empty model. Afterward, they can run the analysis, as explained in Section 4.2.

- `-h, --help`: Show the help message and exit.
- `-i INPUTCSV, --inputcsv INPUTCSV`: Input CSV file - simulated event logs saved from RisQFLan.
- `-idot INPUTDOT, --inputdot INPUTDOT`: Input DOT file - the graphical representation of the procedural part of the RisQFLan model.
- `-ibbt INPUTBBT, --inputbbt INPUTBBT`: Input BBT file - the textual representation of model written in RisQFLan.
- `-o OUTPUT, --output OUTPUT`: Output NAME - diff model between the graphical representation of the procedural part of the RisQFLan model and the mined PM model.

Example:

```
python diff_model.py -i [CSV_file].csv -idot [DOT_file].dot
-ibbt [BBT_file].bbt -o [OUTPUT_file].pdf
```

To conduct the experiments and obtain both the diff model and the new RisQFLan text file with the refined model discussed in the paper:

```
python diff_model.py -i log_RobBank_original.csv -idot
RobBankAttacker.dot -ibbt RobBank.bbt -o-
diff_model_w_ref_script.pdf
```

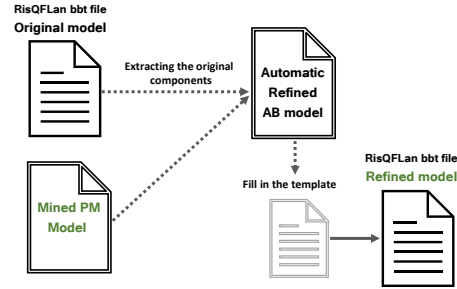


Figure 4: Refinement Method

Similar to the example in the previous section for running only the differential model, in this scenario, the user will be prompted to input the nodes representing the initial assets of the attacker, specifically `FindCode1`, to recreate the same differential model as described in the paper.

## 4.2 ModelsRisQFLan

To run fresh experiments, the user needs to install RisQFLan by following these instructions:


<https://github.com/RisQFLan/RisQFLan/wiki/Install-RisQFLan>.

Once installed the software:

Import the project in RisQFLan:

- Open RisQFLan.
- From the top menu bar click on **File**, and select **Import**.
- Choose **General**, select **Existing Projects into Workspace** and click **Next**.
- Click **Browse...** and locate in the folder *ModelsRisQFLan* the right model to import.
- Tick **Copy projects into workspace** and then hit **Finish**.

Analyze a model

- Open the RisQFLan model of interest from RisQFLan's project explorer.
- Click the button  to run the analysis specified in it.

After completing the analysis, the user is required to transfer the event logs, which have been saved in the workspace of RisQFLan, to the designated *csv*

folder located within the primary *data* directory of the project (as depicted in Figure 2). Then, to run the diff model script, the user should follow the steps outlined in Section 4.1.

### 4.3 Run experiments with new models

Users can also write their own models in RisQFlan by creating a project and then a new RisQFlan file within it. Once the user has defined all the components of the RisQFlan model, they can analyze it as explained in Section 4.2. Before utilizing the `diff_model.py` and `diff_model_auto_refinement.py` scripts, the user must save the DOT, CSV, and BBT files in their respective folders within the *data* folder and then execute one of the two scripts.

## 5 Conclusion

This documentation provides a complete overview of the project, the script's options, and specific instructions for conducting the experiments but also it includes the instructions on how a user can work on their models to validate them using statistical model checking and process mining.

## 6 License

This project is released under the *Apache License Version 2.0*.