A Appendix

A.1 User Manual

This chapter is meant to guide users through setting up necessary environments that are needed to run the software and explain how some key features work.

A.1.1 Required platforms

The software requirements are the following:

- Java 12.0.1 or newer version
- JavaFX 12.0.1 or newer version
- Haskell platform
- OFMC executable - version 2018 or newer

The previously mentioned software can be found on the following websites:

- Java: https://www.oracle.com/technetwork/java/javase/downloads/index.html
- JavaFX: https://gluonhq.com/products/javafx/
- Haskell: https://www.haskell.org/platform/
- OFMC: http://www2.compute.dtu.dk/ samo/

Downloading Java and Haskell will give runnable setup files while JavaFX will be a ZIP file. This ZIP file should be extracted. The *Java Home* folder is an ideal location to place it. The location chosen will be used in setup process described in Appendix A.1.2.

A.1.2 Setup

After installing Java and Haskell along with unzipping the JavaFX folder a couple of environmental variables have to be set. There are separate guides for different operating systems.

A.1.3 Windows

1. Open Control Panel -> System
2. Click the "Advanced" tab
3. Open "Environment Variables"
4. If a variable named "JAVA_HOME" already exists, edit it instead of creating a new one in the next step
5. Click on the "New" button under "System Variables"
6. Add the following and switch out "x.x.x" for the version of downloaded Java and make sure that the path where Java is installed is correct:
   Variable name: JAVA_HOME
   Variable value: C:\Program Files\Java \jdk-x.x.x
7. Click "Ok"
8. Now select "Path" in the "System variables" list and press "Edit"
9. Click "New" and add the following to the new line that appears:
   %JAVA_HOME%\bin
10. Now use the "Move up" button and move the line you just added to the top of the list.
11. Press "Ok"
12. Click on the "New" button under "System Variables"
13. Add the following (**the quotes (") are important!**) and switch out "x.x.x" for the version of downloaded JavaFX and make sure that the path is where JavaFX was un-zipped is correct:
    -Variable name: PATH_TO_FX
    -Variable value: "C:\Program Files\Java\javafx-sdk-x.x.x\lib"
14. Click "Ok"
15. Restart the computer so the new Environment Variables become active
16. Extract ofmc.exe from ofmc20xx.zip/executable for windows. Preferably to the same folder as the GUI is located in.

Now everything should be setup properly and the software is ready to be launched. See Section A.1.5.

A.1.4 Mac OS and Linux

Setup for Mac and Linux is considerably easier than for Windows because all needed options could be put into a single *Makefile*. The *Makefile* was made by Sebastian A. Mödersheim.

1. Edit "Makefile" in any text editor
2. Change the directory for *JAVA_HOME* if it is not the location where Java is installed
3. Change the version of *javafx-sdk* to the version downloaded and if it was not placed in the *JAVA_HOME* directory it should be changed accordingly
4. Extract ofmc from ofmc20xx.zip/executable for mac (or linux). Preferably to the same folder as the GUI is located in.

Now everything should be setup properly and the software is ready to be launched. See Appendix A.1.5.

A.1.5 Running the software

The software cannot be launched simply by double clicking the .jar file. This is because it is built using JavaFX and it needs to be supplied with arguments. To simplify this for the user, a *bat* file was made for Windows users and a *Makefile* for Mac OS and Linux users. Following are instructions how to use those.
NOTE: It is recommended to place the appropriate ofmc runable in the same directory as the *jar* file as well as the *Makefile*.

A.1.6 Windows

If all steps were followed in the Setup process in Appendix A.1.3 the user can simply double click *run.bat* and the software launches.
If nothing happens there are likely some errors occuring when locating JavaFX. The user can edit *run.bat* and add "Pause" in a new line and see the error. It is recommended though to go through the steps in Appendix A.1.3 again and see if any were missed.
NOTE: If the name of *OFMCGUI.jar* has been changed, it is required to change it as well in the *run.bat* file.

A.1.7 Mac OS and Linux

If all paths have been set up correctly in the *Makefile*, the software can simply been run from a terminal by using the command *make*.
If some errors occur that say it cannot locate some of the JavaFX files, the paths are likely not correctly set in the *Makefile*.
NOTE: If the name of *OFMCGUI.jar* has been changed, it is required to change it as well in the *Makefile*.

A.2 How to use OFMCGUI

To demonstrate how OFMCGUI works a few examples will be shown. The examples shown are protocols that can be found in *An AnB Tutorial* [5].

A.2.1 General information

To begin with when the software is launched the user can see the screen shown in Figure A.1.

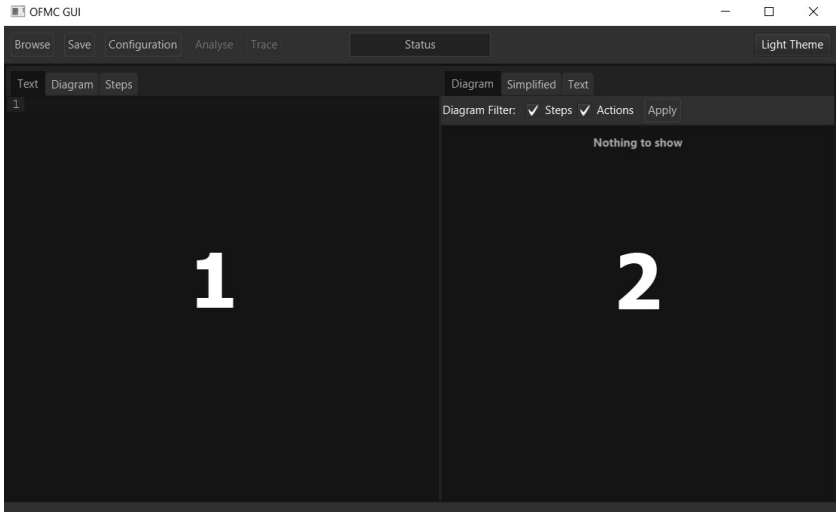

Figure A.1: Startup screen

The screen splits up in two parts labeled as 1 and 2 in Figure A.1. On the screen on the left, labeled 1, the user can either type in his/her protocol that is to be analyzed or load it by using the `Browse` button above. If the syntax is correct the user can then switch between tabs above the text editor and see the diagram that

the protocol produces or the steps that the protocol makes. Both of these extra tabs display the actions taken in the protocol and therefore if there are no actions these tabs will be empty.
The screen on the right, labeled 2, is where the results appear. Given that the syntax is correct, the results can be produced by clicking the `Analyze` button. If the syntax is incorrect it will be displayed in red on the menu bar. This screen has three different tabs, the Diagram, Simplified and Text. If an attack is found, the Diagram tab will produce a diagram that visualizes the attack. Again, if an attack is found, the Simplified tab breaks the attack trace into steps and displays them. If no attack is found both Diagram and Simplified will be empty. If a user prefers the original output of *OFMC* that is supplied in the Text tab.

A.2.2 Example: Analyse

Now to demonstrate how to analyze a protocol, two examples are given.

In Figure A.2 and Figure A.3 the protocol called *First attempt* in *An AnB Tutorial*[5] is analyzed. An attack is found and displayed as a diagram in Figure A.2 and simplified in Figure A.3. The protocol it self can be seen on the left side.

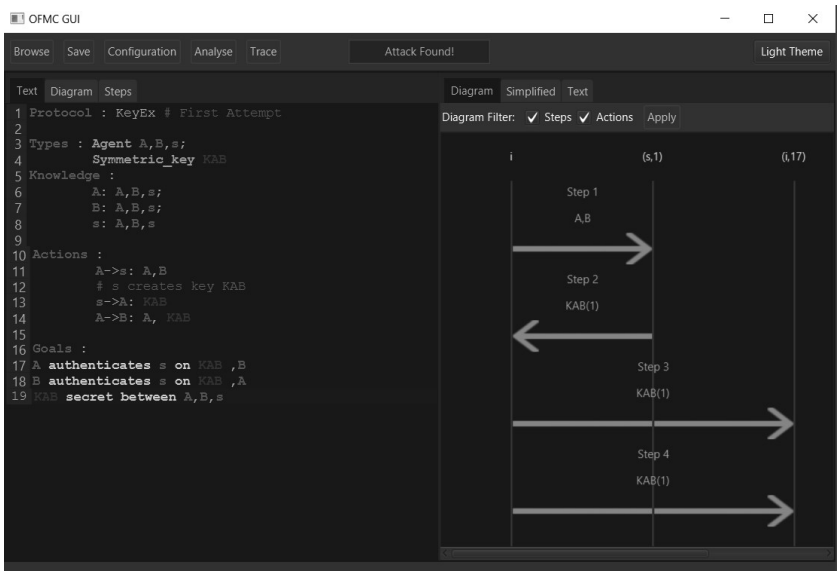

Figure A.2: Analysis: Diagram

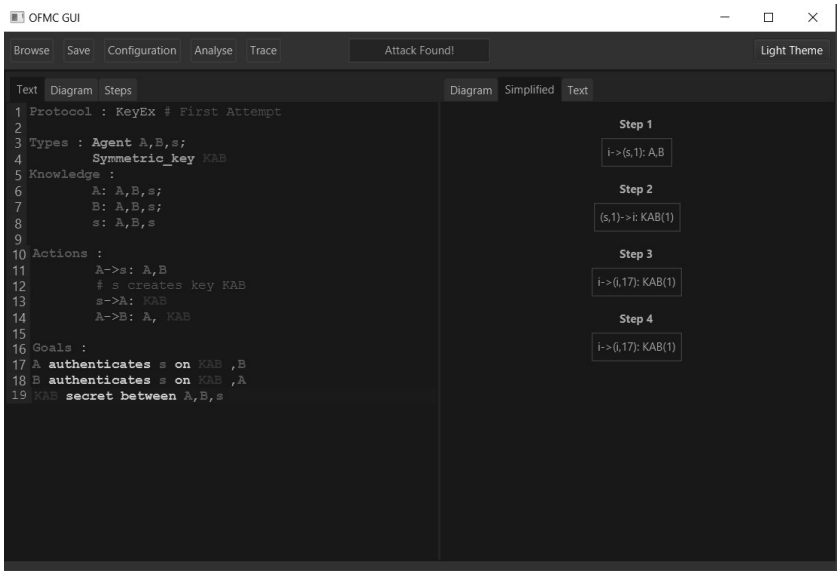

Figure A.3: Analysis: Simplified

The protocol used in this example can be seen here:

```
Protocol : KeyEx # First Attempt

Types : Agent A,B,s;
        Symmetric_key KAB
Knowledge :
        A: A,B,s;
        B: A,B,s;
        s: A,B,s

Actions :
        A->s: A,B
        # s creates key KAB
        s->A: KAB
        A->B: A, KAB

Goals :
A authenticates s on KAB ,B
B authenticates s on KAB ,A
KAB secret between A,B,s
```

The protocol used for the second example, called *Final version* in *An AnB Tutorial* [5], will not lead to an attack. In Figure A.4 nothing appears in the Diagram tab nor the Simplified tab but *No Attack Found!* is displayed in the status window on the menu bar. That implies that *OFMC* did not find an attack on this protocol.

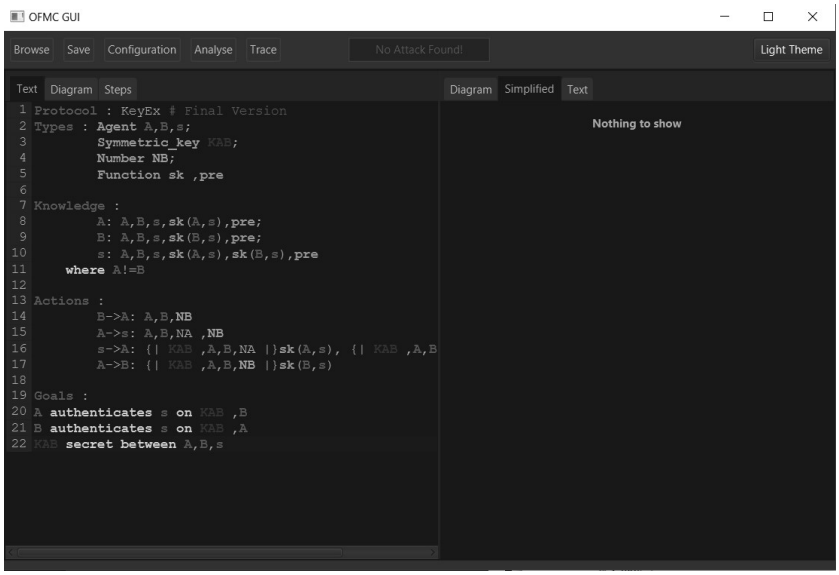

Figure A.4: Analysis: No Attack Found

The protocol used in this example can be seen here:

```
Protocol : KeyEx # Final Version
Types : Agent A,B,s;
        Symmetric_key KAB;
        Number NB;
        Function sk ,pre

Knowledge :
        A: A,B,s,sk(A,s),pre;
        B: A,B,s,sk(B,s),pre;
        s: A,B,s,sk(A,s),sk(B,s),pre
    where A!=B

Actions :
        B->A: A,B,NB
        A->s: A,B,NA ,NB
        s->A: {| KAB ,A,B,NA |}sk(A,s), {| KAB ,A,B,NB |} sk(B,s)
        A->B: {| KAB ,A,B,NB |}sk(B,s)

Goals :
A authenticates s on KAB ,B
B authenticates s on KAB ,A
KAB secret between A,B,s
```

A.2.3 Example: Trace

The possibility to use the Trace option of OFMC is also available. If the syntax of the protocol is correct, the user should be able to press the *Trace* button. When pressed, a new tab opens on the right side. This tab contains the options that are available. When the user clicks on one of the options he/she will be presented with new set of options that are a result of selecting the previous one. The user can navigate back and forth through the previous options chosen and select different ones if needed and continue from there. These features can be seen in Figure A.5. The protocol used is the *First Attempt* protocol showcased in Section A.2.2.

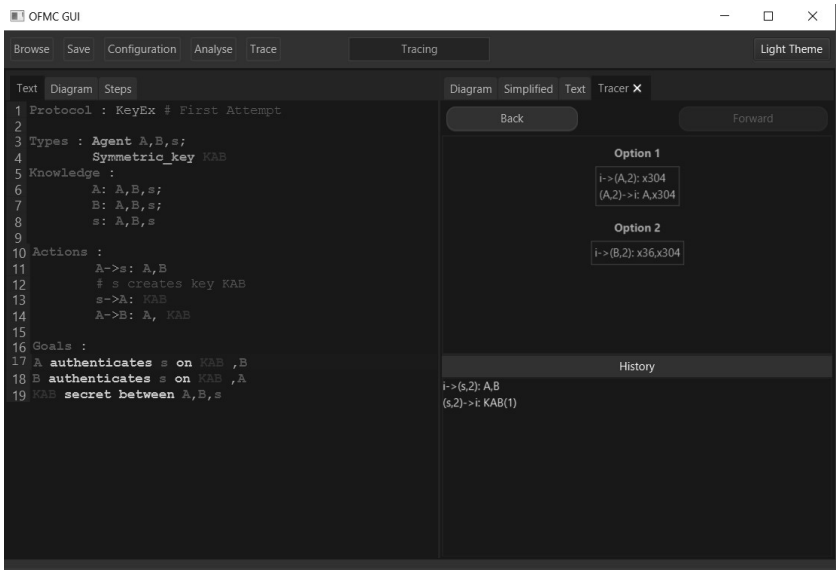

Figure A.5: Trace: No Attack Found

References

[1] Sebastian Mödersheim. *Open Source Fixedpoint Model Checker - Short Manual*. IBM Research Lab, Säumerstrasse 4, 8803 Ruschlikon, Switzerland, November 2009.

[2] Gluon. *JavaFX*. Available at: https://gluonhq.com/products/javafx/, 2019. *Located 26-05-2019*.

[3] Tomas Mikula et al. *RichTextFX*. Available at: https://github.com/FXMisc/RichTextFX, 2019. *Located 27-05-2019*.

[4] Terence Parr. *About The ANTLR Parser Generator*. Available at: https://www.antlr.org/about.html, 2014. *Located 27-05-2019*.

[5] Sebastian Mödersheim. *An AnB Tutorial*. DTU Informatics, September 2013.