

## 4.3 Manual

**Installation** The CD-Rom attached to this thesis contains the AQL-System as an executable .jar file as well as the Eclipse<sup>4</sup> project used to built it. Note that Java 1.8 or newer is required in order to run and build the AQL-System. However, no preprocessor and only one analysis tool, namely PAndA<sup>2</sup>, is shipped with the tool. While PAndA<sup>2</sup> can be found in the "Tools" directory, FlowDroid<sup>5</sup> and IC3<sup>6</sup> can be obtained online. In the "Tools" directory several bash and batch scripts can be found next to PAndA<sup>2</sup>. These can be used to execute FlowDroid, IC3 and PAndA<sup>2</sup> on Linux or FlowDroid and PAndA<sup>2</sup> Windows. Furthermore, these scripts are included in a exemplary configurations (see "Examples" directory). Once all tools that should be used are installed, the configuration file ("config.xml") has to be adapted. How to adapt the configuration is explained in Section 4.1.1. Thereafter the AQL-System is ready to be used.

**Usage** The AQL-System can be accessed from the command-line through the following command:

```
java -jar /path/to/AQLSystem.jar
```

Furthermore, the parameters shown in Table 4.1 may be attached. The following command, for example, queries the system in order to ask for all permissions used by the SIMApp:

```
java -jar /path/to/AQLSystem.jar -query "Permissions IN App('SIMApp.apk')
?" -o "result.xml"
```

The answer to this query is stored in the file "result.xml". It is also possible to use the AQL-System inside another Java project. Therefore, add the AQLSystem.jar as library and use the following three instructions to import the `System` class and to invoke a query:

```
import de.foellix.aql.system.System;

final System system = new System();
system.query("Permissions IN App('SIMApp.apk') ?");
```

Furthermore, the method `getAnswerReceivers()` of a `System` class object can be used to obtain and edit a list of objects<sup>7</sup>. Each object in the list receives all answers computed by the associated system.

A screenshot of the AQL-System's graphical user interface (GUI) is shown in Figure 4.3. The GUI consists of two parts, namely the *Editor* to formulate AQL-

<sup>4</sup>Eclipse IDE: <https://www.eclipse.org> - 04/20/2017

<sup>5</sup><https://github.com/secure-software-engineering/soot-infoflow-android/wiki> - 04/20/2017

<sup>6</sup><http://siis.cse.psu.edu/ic3/> - 04/20/2017

<sup>7</sup>These objects have to implement the interface `IAnswerAvailable`

Parameter	Meaning
-help, -h, -?, -man, -manpage	Outputs a very brief manual, which contains a list of all available parameters.
-query "X", -q "X"	This parameter is used to assign the AQL-Query. X refers to this query.
-config "X", -cfg "X", -c "X"	By default the config.xml file in the tool's directory is used as configuration. With this parameter a different configuration file can be chosen. X has to reference the path to and the configuration file itself.
-output "X", -out "X", -o "X"	The answer to a query is automatically saved in the "answers" directory. This parameter can be used to store it in a second file. X has to define this file by path and filename.
-preferexecute, -pe	This parameter is used to force the execution of analysis tools even if a similar question has been asked before.
-timeout "X"s/m/h, -t "X"s/m/h	With this parameter the maximum execution time of each tool can be set. If it expires the tool's execution is aborted. X refers to this time in seconds (e.g. 10s), minutes or hours.
-debug "X", -d "X"	The output generated during the execution of this tool can be set to different levels. X may be set to: "error", "warning", "normal", "debug", "detailed" (ascending precision from left to right). By default it is set to "normal".
-gui	If this or no parameters at all are provided the graphical user interface is started. It allows to formulate queries and display answers in a handy way.

Table 4.1: Parameters

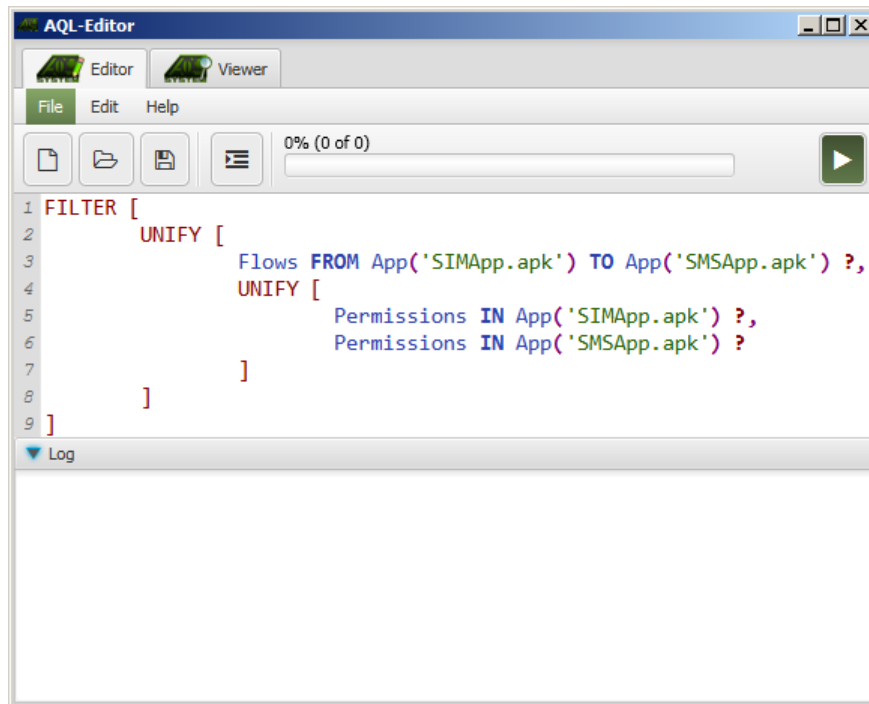


Figure 4.3: Screenshot of AQL-System's GUI

Queries and the *Viewer* to view AQL-Answers. The Editor is capable of highlighting and auto-formatting AQL-Queries as well as asking them by clicking the play-button at the top right corner. Once a query is asked, the bottom area titled with "log" is used to display status messages including errors and warnings. The progressbar at the top shows how many preprocessors and analysis tools have to be executed until the query can be answered. The viewer is not visible in the screenshot. It offers two possibilities to display an AQL-Answer: On the one hand, a textual view with syntax highlighting can be used to directly view and edit an .xml file representing an AQL-Answer. On the other hand, a graphical view shows the flows, permissions, intent-sources and -sinks in a single graph. Thereby the graphical view provides a better overview. An example of such a graph is drawn in Figure 3.3 on Page 53 and was explained at the end of Section 3.3.3.