CST 412

Assignment 4

Programming language. *For this assignment, you can choose any programming language (Python is recommended).*

Problem description:
You will work in a group to implement a framework to enforce and analyze the ABAC model that we discussed in class.

Framework features:

1. The framework should provide data structures to represent ABAC policies following the policy language that we discussed in class.

2. The framework should provide a feature to parse ABAC policies from input files (`.abac` files) and store them in the data structures designed in 1. A detailed description of the `.abac` files is provided below.

3. The framework should provide a feature to check if a request is permitted or denied. A request contains a (1) `sub_id`, (2) `res_id`, and (3) `a`, specifying that the subject with `sub_id` is trying to perform the action `a` on the resource with `res_id`. The request is permitted if at least one of the rules grants the permission, and is denied otherwise.

4. The framework should provide a policy coverage analysis feature that calculates the number of authorizations (permissions) each rule covers and identifies the attributes referenced within each rule. This analysis would produce a heatmap with rules on one axis and attributes on the other, where the intensity indicates the volume of authorizations (permissions) covered by a rule that references a particular attribute.

Provided input file:

- `university.abac`: the sample university ABAC policy input file.
- `healthcare.abac`: the sample health care ABAC policy input file.
- `university-requests.txt`: sample text file containing access requests for the university sample policy.
- `healthcare-requests.txt`: sample text file containing access requests for the healthcare sample policy.

A detailed description of the `.abac` files can be found here, and a detailed description of the sample access requests file is available here.

Requirements:

- Your framework should provide a main (driver) program named "`myabac`". The program should provide two (2) running options:

  - `-e`: This option requires two additional arguments: the first argument specifies the input ABAC policy file (.abac), and the second argument specifies the request file. For example, if you write your program in Python, the program should be able to run using the command:

    ```
    python3 myabac.py -e university.abac university-requests.txt
    ```

    Expected output: your program should print out the access decision for each request as described in Assignment 3. Access decision can be “Permit/Deny”, “True/False”, or “Yes/No”.

  - `-a`: This option requires one argument specifying the input ABAC policy file (.abac). For example, if you write your program in Python, the program should be able to run using the command:

    ```
    python3 myabac.py -a university.abac
    ```

    Expected output: your program should output the heatmap of the coverage analysis explained above for the input policy.

- You must provide a document with instructions on how to compile and run your program with command-line commands. The instructions should not depend on any use of an IDE.

Contributions

Your group is responsible for assessing the performance of each member using the provided evaluation form, which can be downloaded here as an .xlsx file. After assigning ratings to the spreadsheet following the provided instructions in the file, please save the completed document for the submission. *It's crucial to gather feedback from all members during the evaluation process. Additionally, review the completed spreadsheet with all members before submission for transparency and consensus.*

Submission Instructions:

Only one submission per team. Please submit a *zip file* containing the following:

1. **Source Code**: Include only the source code files for your framework. You do not need to submit any input files.
2. **Instructions Document**: The document with instructions on how to compile and run your program with command-line commands as explained in the Requirements section.
3. **Heatmaps**: Include two heatmaps generated by your framework, showing the results of the coverage analysis on the provided ABAC policy files: `university.abac` and `healthcare.abac`.
4. **Team Evaluation**: The completed team evaluation file as described in the Contributions section.

Grading Procedure:

For grading, we will use a script to download your zip file and extract the contents into a folder containing our test input files. The script will then automatically run your program using the two options described in the previous section, with the test input files as arguments. *Please note that the source code will be stored in the same directory with the test input files and testing will also be performed from this directory.*

Grading Rubric:

Task	**Point**
Access requests evaluation.	40
Coverage analysis.	40
Program compiled and ran successfully and well-commented code	20
Total	100