Report 1: Characterizing and understanding security risks through Security-Aware Mutation Testing of security configuration in RESTful APIs

Carlos Andres Delgado Saavedra
carlos.andres.delgado@correounivalle.edu.co Advisors:
Jesus A. Aranda, Universidad del Valle
Gills Perrouin, University of Namur, James Ortiz, Universite of Namur

November 24, 2024

1 Introduction

The focus of this report is the exploration and development of the `mutpy` library as part of a doctoral thesis. The structure of the `mutpy` library is as follows:

```
|-- mutpy
|   |-- codegen.py
|   |-- commandline.py
|   |-- controller.py
|   |-- coverage.py
|   |-- __init__.py
|   |-- operators
|   |   |-- arithmetic.py
|   |   |-- base.py
|   |   |-- decorator.py
|   |   |-- exception.py
|   |   |-- inheritance.py
|   |   |-- __init__.py
|   |   |-- logical.py
|   |   |-- loop.py
|   |   |__ misc.py
|   |-- templates
|   |   |-- base.html
|   |   |-- detail.html
|   |   |__ index.html
|   |-- termcolor.py
|   |-- test
|   |   |-- __init__.py
|   |   |-- test_commandline.py
|   |   |-- test_controller.py
|   |   |-- test_coverage.py
|   |   |-- test_operators.py
|   |   |-- test_runners.py
|   |   |-- test_utils.py
|   |   |-- test_views.py
|   |   |__ utils.py
|   |-- test_runners
|   |   |-- base.py
|   |   |-- __init__.py
|   |   |-- pytest_runner.py
|   |   |__ unittest_runner.py
|   |-- utils.py
|   |__ views.py
```

The operators are specified in the `operators` folder, which receives the abstract syntax tree (AST).

2 Installation of the library

I created a fork of this library here: https://github.com/cardel/mutpy, I detected some problems about the compability, so I need to create a Docker container:

```
FROM python:3.9
COPY . usr/src/app
WORKDIR /usr/src/app
RUN apt-get update && apt-get install -y git
RUN pip install pip --upgrade
RUN pip install -r requirements.txt
RUN sh setup.sh
RUN python -m unittest discover tests
ENTRYPOINT sh tests.sh
```

And I needed to add some scripts:

Setup.sh

```
git clone https://github.com/cardel/mutpy.git
cd mutpy && python setup.py install
```

test.sh

```
mut.py --target run.py --unit-test tests/
    test_app.py -m
```

3 Exploration of Flask Configurations

The current configurations of Flask are loaded into the main variable as follows:

```
print("Current Flask configurations:")
for key, value in app.config.items():
    print(f"{key}: {value}")
app.run(port=3000, debug=True)
```

Where `app` is the main variable. Basic Flask configurations include:

```
DEBUG: False
TESTING: False
PROPAGATE_EXCEPTIONS: None
SECRET_KEY: None
PERMANENT_SESSION_LIFETIME: 31 days, 0:00:00
USE_X_SENDFILE: False
SERVER_NAME: None
APPLICATION_ROOT: /
SESSION_COOKIE_NAME: session
SESSION_COOKIE_DOMAIN: None
SESSION_COOKIE_PATH: None
SESSION_COOKIE_HTTPONLY: True
SESSION_COOKIE_SECURE: False
SESSION_COOKIE_SAMESITE: None
SESSION_REFRESH_EACH_REQUEST: True
MAX_CONTENT_LENGTH: None
SEND_FILE_MAX_AGE_DEFAULT: None
TRAP_BAD_REQUEST_ERRORS: None
TRAP_HTTP_EXCEPTIONS: False
EXPLAIN_TEMPLATE_LOADING: False
PREFERRED_URL_SCHEME: http
TEMPLATES_AUTO_RELOAD: None
MAX_COOKIE_SIZE: 4093
```

Default security settings might include:

- `SESSION_COOKIE_SECURE: True`
- `SESSION_COOKIE_SAMESITE: 'Strict'`
- `CSRF_ENABLED: True`
- `X_FRAME_OPTIONS: 'SAMEORIGIN'`

4 Development Environment and Docker Setup

Due to compatibility issues, a Docker container was necessary. The setup is as follows:

```
FROM python:3.9
COPY . /usr/src/app
WORKDIR /usr/src/app
RUN apt-get update && apt-get install -y git
RUN pip install pip --upgrade
RUN pip install -r requirements.txt
RUN sh script.sh
RUN python -m unittest discover tests
ENTRYPOINT sh tests.sh
```

The `script.sh` contains:

```
#!/bin/bash
git clone https://github.com/cardel/mutpy.git
cd mutpy && python setup.py install
```

`tests.sh` runs:

```
mut.py --target run.py --unit-test tests/
    test_app.py -m
```

5 Mutation Operators

Mutation operators interact with the AST of the file to be mutated, generating a mutant. For example:

```
def mutate_FloorDiv_to_Div(self, node):
    if self.should_mutate(node):
        return ast.Div()
    raise MutationResign()

def mutate_FloorDiv_to_Mult(self, node):
```

```
    if self.should_mutate(node):
        return ast.Mult()
    raise MutationResign()
```

An example for disabling CORS:

```
app.config["ALLOW HEADERS"] = "*"
```

The exploration and development with `mutpy` and modification to Flask applications are integral parts of this doctoral thesis progress.