UGA **Grenoble INP**
UFR IM2AG **Ensimag**

M2 CySec – Software Security

Code Analysis with CodeQL

The objective of this lab is to discover some functionalities of CodeQL for code security purposes . There is no report to deliver at the end of the session, but you should get some basic understanding on:

- how codeQL works;
- which kind of facilities does it bring;
- how to read/write basic requests and interpret their results in code examples.

At the end of the day you should have some clear comparison elements between all the code analysis techniques we saw these last week: fuzzing, symbolic execution, value analysis and semantic pattern recognition ...

Setup

We are going to use VSCode, with the CodeQL extension, on the Ensimag computers.

1. Launch Vscode (code) and chech that the CodeQL extension is already available (it should be the case; if not install it …).
2. Download the Workspace available on the Moodle web page (and extract its content in a directory of your choice)
3. Open this workspace on VSCode
   VSCode: *File → Open Workspace from File*

choosing the file `vscode-codeql-starter/vscode-codeql-starter.code-workspace`

You can now open the CodeQL extension …

Running some basic queries on a « large » code base

Within CodeQL, select *Download a database from github* and enter this URL:
https://github.com/protocolbuffers/protobuf

You can then execute (and understand !) some of the simple queries proposed on this page:
https://codeql.github.com/docs/codeql-language-guides/basic-query-for-cpp-code/

You can then have a look this page:
https://codeql.github.com/docs/codeql-language-guides/expressions-types-and-statements-in-cpp/

Then, write a request allowing to find all the occurrences of some specific functions (e.g., malloc and free), using this page:

https://codeql.github.com/docs/codeql-language-guides/functions-in-cpp/#

And finally, write a request to find all occurrences of pointer assignments to NULL (if any?) ...

Running more specific queries on your own code

Have a look to the following page to better understand some of the facilities offered by CodeQL for (C/C++) code pattern recognition :

Data-Flow requests:
https://codeql.github.com/docs/codeql-language-guides/analyzing-data-flow-in-cpp/

Various general request examples:
https://codeql.github.com/codeql-query-help/cpp/

And the whole set of requests already available in your workspace:
https://github.com/github/codeql/tree/main/cpp/ql/src
(see in particular the *Critical* and *Security* folders)

The goal is now to freely check (and possibly change/extend) some of these requests on small programs of your own …

To do so you need to create a directory to put your examples (you can use the my-examples one provided on Moodle) and to build a database (since CodeQL does not work from raw code) using the following command:

```
/user/7/mounlaur/CodeQL/codeql/codeql database create –language=cpp \
--search-path=/user/7/mounlaur/CodeQL/codeql/cpp/ my-examples-db --overwrite \
--command='gcc ex1.c -o ex1' --source-root my-database/
```

You can then load from VSCode the fresh database my-database-db and run some requests against it (**updating the C file accordingly, and removing then reloading the database each time you change it ...**)

In particular, you should focus on Critical and Security requests. You can for instance have a detailed look on the CWE list and see which of your examples are caught or not by these provided requests ...