TP - Assisted Code Analysis

revision 1.2

Aim

We will use the same codes as last TD, but this time we focus on trying to find the issues with both static and dynamic analysis tools.

For each tool :

1. Install it
2. Run it on the codes
3. See which issues it detects
4. Try to understand why it does/does not find the issues

Static Analysis

GCC Warnings

Install : `apt install gcc g++` *(gcc and g++ version at least 13)*
Run (C) : `gcc -std=gnu2x -O2 -Wall -Wextra file.c`
Run (C++) : `g++ -std=gnu++2b -O2 -Wall -Wextra file.cpp`

Clang Warnings

Install : `apt install clang`
Run (C) : `clang -std=gnu2x -O2 -Wall -Wextra file.c`
Run (C++) : `clang++ -std=gnu++2b -O2 -Wall -Wextra file.cpp`

cppcheck

Install : `apt install cppcheck`
Run : for you to find out. Read the documentation

GCC Analyzer

Install : already done above
Run (C) : `gcc -std=gnu2x -O2 -fanalyzer file.c`
Run (C++) : `g++ -std=gnu++2b -O2 -fanalyzer file.cpp`

Clang Analyzer

Install : `apt install clang-tools`
Run (C) : `scan-build clang -std=c2x -O2 -c -w file.c`
Run (C++) : `scan-build clang++ -std=gnu++2b -O2 -c -w file.cpp`

Dynamic Analysis

GCC Sanitizers

Install : already done above
Run (C) : `gcc -std=gnu2x -O2 -w -fsanitize=XXX file.c`
Run (C++) : `g++ -std=gnu++2b -O2 -w -fsanitize=XXX file.cpp`
Note : replace XXX

Clang Sanitizers

Install : already done above
Run : for you to find out. Read the documentation

Valgrind

Install : `apt install valgrind`
Run : `valgrind ./file`

tis-interpreter (optional)

Install and run : follow instructions on https://github.com/TrustInSoft/tis-interpreter

References

Useful tools list : https://github.com/analysis-tools-dev/static-analysis and https://github.com/analysis-tools-dev/dynamic-analysis.
Codes snippets are adapted from https://github.com/atxsinn3r/VulnCases/.