Git integration of cppcheck for static code analysis

Can we ever imagine sitting back and manually reading each line of codes to find flaws? To ease our work, several types of **static analysis tools** are available in the market which helps to analyze the code during the development and detect fatal defects early in the SDLC phase. Such defects can be eliminated before the code is actually pushed for functional QA. **A defect found later are always expensive to fix.**

There are several open source static code Analysis(SCA) tools are available in the market, like:

1. cppcheck
2. oclint
3. clang
4. cppclean
5. flint

Here, **cppcheck** is better option among others, because it has more features, lower execution time as well as it supports integration with many tools compared to other open source static code analyzers.

cppcheck user manual for reference:

cppcheck_manual.pdf

Cppcheck integration with the FirstAlert GVA Git

Objective:

As a part of CI/CD activity, Integrate the cppcheck tool with the GIT. As of now, we can go forward with below *limitations*:

1. We can skip to fix the current cppcheck defects - will fix later on.
2. Comparing to current defects as baseline, No newer defects should be occurred.
3. Mediatek/ SDK can be skipped from cppcheck execution.

Prerequisites:

1. SHELL SCRIPTING:
2. GIT HOOKS:

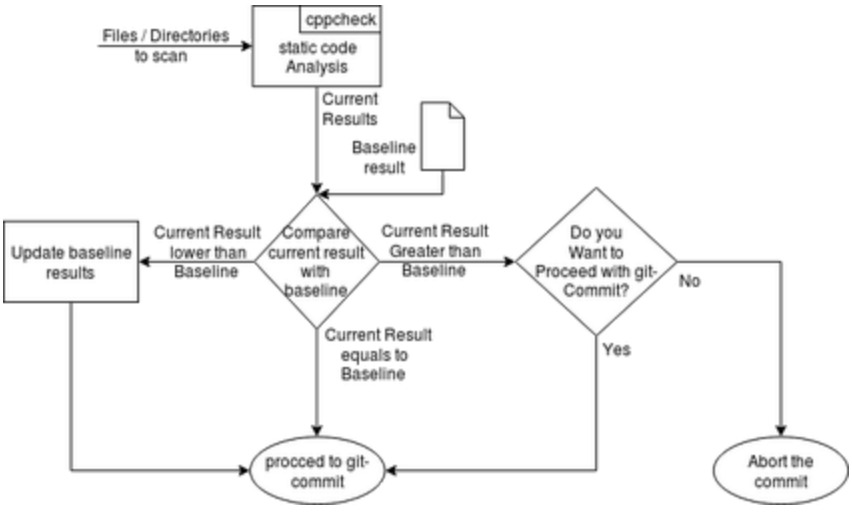
   Git hooks are nothing but a shell scripts that Git executes before or after events such as: commit, push, and receive. Git hooks are a built-in feature - no need to download anything. Git hooks are run locally. These hook scripts are only limited by a developer's imagination.

   For more knowledge on Git hooks, please visit to https://githooks.com

Implementation:

As per our requirement, we need to customize a pre-commit hook (*.git/hooks/pre-commit*) to achieve our requirement.

IMPLEMENTATION FLOW :



Pseudo-code of pre-commit hook to achieve the requirement:

```
!/bin/sh
#
# This script enables pre-commit cppcheck validation and
# warn user if newer cppcheck defects are detected.
#

# Check whether cppcheck is installed or not

if [ cppcheck_installed ]; then

    # Execute cppcheck on change-log/directory
    # Get current cppcheck defects
    # Get baseline defects

    if [ current_defects > baseline_defects ]; then
        echo "Newer defects detected"
        echo "Do you want to continue with the git-commit [y/n]
?"
        # Proceed with git-commit on Y, else
        # Abort the commit
    elif [ current_defects < baseline_defects ]; then
        # Update the baseline result and proceed with git-commit
    else
        # No new defects found, go forward with git-commit
    fi
fi
```

CONFIGURATION AND DIRECTORY-STRUCTURE:

Git does not provide a luxury to push the files under **.git/** directory on the main repository. So, to share this integration with team, we must need to find/choose a way with minimal configuration efforts required.

For FirstAlert GVA, we shall follow below way to share the cppcheck git hook:

1. Make the pre-commit hooks under a separate *.githooks/* directory for each module / sub-module
2. Make a shell script to copy the *.githooks/* directory to respected module git directory
3. Add an option in Makefile to init the cppcheck configuration and execute the shell script in the configuration

For more general ways to share the git hooks with the team, one can refer this link.

```
gva/                                          (( Copied to ))                                        gva/.git/
|                                                                                                           |
|___FirstAlert/.githooks/pre-commit                 --------->                                hooks/pre-commit__|
        |                                                                                                   |
        |__acs/.githooks/pre-commit                 --------->              modules/FirstAlert/acs/hooks/pre-commit__|
        |                                                                                                   |
        |__scs/.githooks/pre-commit                 --------->              modules/FirstAlert/scs/hooks/pre-commit__|
        |                                                                                                   |
        |__sws/.githooks/pre-commit                 --------->              modules/FirstAlert/sws/hooks/pre-commit__|
        |                                                                                                   |
        |__cas/.githooks/pre-commit                 --------->              modules/FirstAlert/cas/hooks/pre-commit__|
        |                                                                                                   |
        |__superdog_gva/.githooks/pre-commit        --------->        modules/FirstAlert/superdog_gva/hooks/pre-commit__|
        |                                                                                                   |
        |__CommonFirmware/.githooks/pre-commit      --------->      modules/FirstAlert/CommonFirmware/hooks/pre-commit__|
        |                                                                                                   |
        |__OnelinkMasterControl/.githooks/pre-commit --------->  modules/FirstAlert/OnelinkMasterControl/hooks/pre-commit__|
        |                                                                                                   |
        |__OnelinkLEDAnimation/.githooks/pre-commit  --------->   modules/FirstAlert/OnelinkLEDAnimation/hooks/pre-commit__|
```

Post-Development :

We have integrated the cppcheck into the FirstAlert GVA with respect to above design. One can refer below PRs for more insights:

- https://github.com/jardenss/gva/pull/226
- https://github.com/jardenss/CommonFirmware/pull/28

- https://github.com/jardenss/OnelinkLEDAnimation/pull/3
- https://github.com/jardenss/OnelinkMasterControl/pull/66
- https://github.com/jardenss/ACS/pull/42
- https://github.com/jardenss/CAS/pull/42
- https://github.com/jardenss/SCS/pull/51
- https://github.com/jardenss/superdog_gva/pull/7
- https://github.com/jardenss/SWS/pull/21

One can use below commands for configuring cppcheck in GVA:

`make cppcheck-init`	`This command will install the cppcheck tool (if not found), as well as it will copy the customized pre-commit hooks into the relative .git/ directories.`
`make cppcheck-deinit`	`In case of any misdeed, one can de-configure the cppcheck validation using this command.`

SOME SCREEN-SORTS OF THE CPPCHECK IN ACTION:

- When defects on development branch greater than baseline defects:

```
                    :~/Desktop/gva-home/repos/repo2.0/gva/FirstAlert/cas(feature/2018.12.21_FW-789_cppcheck_static_code_analysis_integration) o git c
ommit
[info] cppcheck static code analysis triggered
[info] Cppcheck static code analysis finished, log-file : cppcheck_cas_current_defects.log
[info] Checking current defect results with baseline defect results...
[info] Baseline defects count : 4
[info] Current defects count : 5
[error] Newer defects detected
Do you still want to procced with git-commit? [y/n]: n
[info] aborting git commit
```

- When defects on development branch less than baseline results

```
                    :~/Desktop/gva-home/repos/repo2.0/gva/FirstAlert/cas(feature/2018.12.21_FW-789_cppcheck_static_code_analysis_integration) o git c
ommit
[info] cppcheck static code analysis triggered
[info] Cppcheck static code analysis finished, log-file : cppcheck_cas_current_defects.log
[info] Checking current defect results with baseline defect results...
[info] Baseline defects count : 4
[info] Current defects count : 3
[info] Baseline result is needs to be updated
[info] Baseline result is updated and staged for commit
[warning] Do not checkout the baseline result file : .githooks/.cppcheck_cas_baseline_defects.log
[info] Press any key to continue...
```

- When defects on development branch in sync with baseline defects

```
                    :~/Desktop/gva-home/repos/repo2.0/gva/FirstAlert/cas(feature/2018.12.21_FW-789_cppcheck_static_code_analysis_integration) o git c
ommit
[info] cppcheck static code analysis triggered
[info] Cppcheck static code analysis finished, log-file : cppcheck_cas_current_defects.log
[info] Checking current defect results with baseline defect results...
[info] Baseline defects count : 4
[info] Current defects count : 4
[success] cppcheck current defects are in sync with baseline defects
[info] Press any key to continue...
```

HOW TO CHECK WHICH NEWER DETECTED DEFECT:

One can compare the baseline defects with the development branch defects to get the newer detected defect.

e.g. meld FirstAlert/cas/cppcheck_cas_current_defects.log FirstAlert/cas/.githooks/.cppcheck_cas_baseline_defects.log

LIMITATIONS OF THE CURRENT INTEGRATION:

- **No check on File type** : We are triggering the cppcheck on any change of the current branch to be committed - We are not checking file type.
- **Trigger on branch** : We are triggering the cppcheck on whole branch irrespective to any specific change-log
- **Trigger for specific directories** : As of now, we are checking defects for only **FirstAlert/** directory.

FUTURE WORK:

- One can add mechanism to generate log for only newly detected defect/s.
- One can run cppcheck for selected file-types or only on change-log
- Once all the baseline defects get resolved, one can update the hooks