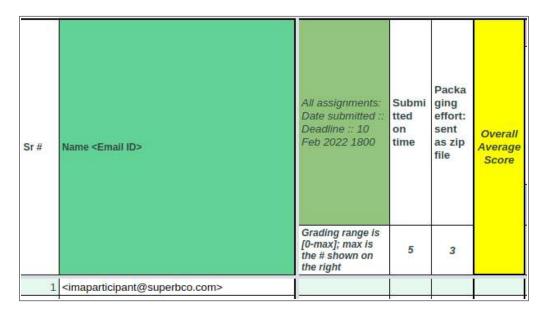
## Linux System Programming course (c) Kaiwan N Billimoria, kaiwanTECH

## Sample Grading Template for assignments issued to participants

## **Instructions**

- Keep each assignment in a separate directory with it's own Makefile
- (Read remaining points below).
  - 1. First portion of the LSP Grading Template:



2. Second portion of the LSP Grading Template:

Assignment 1: 1_sleepsafe											Weightage	1.25	
Builds cleanly (no err/wrn or only warnings)	Global Makefile [with CFLAGS, clean, indent, dbg, install targets, etc]	source license	code readability: k code style, indentation / comments [checkpatch.pl -f+1]	8.5	/sis checks	dynamic analysis performed		documentatio n	working / correctness / desired result (+algo, code,etc)	Optional: Test case(s) [1]	score	%age	Instructor Comments
make_score	makefile_tgt_sc ore	license_score	semiauto [checkpatch_sc ore]	flawfinder_scor e	cppcheck_scor e	asan_score	valgrind_score	doc_score	manual				
5	9	1	5	2	2	4	3	2	10	3	57.5	100%	
					2						0	0%	

- 3. You MUST use **the 'better' Makefile** for each assignment. (This will automatically ensure you get marks for the second column above)
- 4. Next, **ensure you actually run the targets** in the Makefile; type 'make help' to see all of them; use them, fix errors / warnings and retry until it's clean (as far as possible; sometimes, tools like flawfinder can emit false positives that you can ignore). Minimally, ensure the following:
  - i. **make prod** no errors, no warnings (as far as possible)
  - ii. **make debug** no errors, no warnings (as far as possible)
  - iii. code-style
    - 1. make indent
    - 2. **make checkpatch** fix any and all errors, warnings as far as possible
- (c) kaiwanTECH, Kaiwan N Billimoria

- iv. **make sa** static analysers (flawfinder, cppcheck) fix any and all errors, warnings as far as possible
- v. **make valgrind** fix any and all errors, warnings as far as possible
- vi. make san ASAN, UBSAN, MSAN; fix any and all errors, warnings as far as possible
- vii. **make covg** check the code coverage; objective is to write test cases until 100% coverage is achieved

## A shortcut: doing

make test

runs pretty much *all* the above targets. So, you can save all of it (like a report) to a file by doing:

make -i test >out 2>&1 # to save all output to a file "out"

-i = --ignore-errors