Design Document - ASSIGNMENT 2 - Scheduling
CMPS 111 - Darrell Long - Fall 18
Code by Zac Plante(zplante)

Changes made to FreeBSD source code

subr_param.c

added two new variables *priority_sched* and *splatter_sched.* These can be tuned to switch between the 4 cases. This way you can switch scheduling at runtime for easier testing and implementation. 0 means the mode is inactive and 1 means it is active. Switch between them using *sysctl.*

runq.h

added two new functions *runq_add_splatter()* and *add_priority().* These are used to implement splatter and priority scheduling respectively

kern.switch.c

defined the two functions above. also added if statements to *runq_add()* and *runq_add_pri()* to implement priority scheduling if its on.

sched_ule.c

edited the function *tdq_add()* to call either *runq_add()* or *runq_add_splatter()* depending on which is needed. Also doesn't utilize splatter if its a kernel thread.

How Splatter Works

Splatter scheduling code looks almost the same as the regular *runq_add()* but instead of pulling priority from the thread it uses rand() to find its place in the array

How Priority Works

Threads are added to the TAILQ based on they're priority. I utilize the fact that freeBSD already takes from the head to minimize the amount of new code I needed to write