XV6 Scheduling Project Roadmap & Guide

■ Roadmap for Your Project

- Set Up Environment: Install dependencies, clone xv6 source code, run it in QEMU.
- Understand Baseline (Round Robin in xv6): Study how default scheduler works in proc.c.
- Implement FCFS: Add arrival_time, modify scheduler for FIFO behavior, disable preemption.
- Implement Lottery Scheduling: Add tickets, settickets() syscall, implement weighted random selection.
- Scheduler Policy Switch: Add scheduling_policy flag, implement setpolicy() syscall.
- Design Test Programs: CPU-bound, I/O-bound, and Mixed workloads.
- Measure Performance: Collect turnaround, waiting, response times, throughput, fairness.
- Prepare Final Report: Tables + graphs, analysis of tradeoffs, fairness, starvation issues.

■ How to Implement Each Part

- Modify proc.c → scheduling logic.
- Modify proc.h → add arrival_time, tickets fields.
- Modify sysproc.c, syscall.c, user.h, usys.S → new syscalls.
- Implement one change at a time, rebuild with make qemu, test correctness.

■ Designing the Test Bench

- CPU-bound test: Infinite loop computing.
- I/O-bound test: Repeated sleep() calls.
- Mixed test: Combination of both.
- Run tests with different schedulers using setpolicy().
- Collect metrics manually or extend ps command.

■ What is QEMU?

- QEMU is a machine emulator.
- In this project: Simulates x86 hardware to run xv6 safely.
- Used to run and debug modified scheduling algorithms.

■ Installing QEMU on Ubuntu

- Run: sudo apt-get update && sudo apt-get install --yes build-essential git qemu-system-x86
- Check installation: qemu-system-i386 --version or qemu-system-x86_64 --version

■■ Running xv6 in QEMU

- Clone: git clone https://github.com/mit-pdos/xv6-public
- Compile: make qemu
- Run: xv6 shell appears inside QEMU.