Final Year Project

1 Idea

Combine the resources of two separate machines into one. Each user-land process can access the resources of either machine.

2 How it will be achieved

I plan to use an existing open-source micro-kernel operating system (Redox).

I would change the process queue to be shared between the two kernels. When a kernel allows a process to run, it'll choose a process from this shared queue.

Which kernel should run which process? If one kernel has a resource that the process wants, maybe that kernel should run the process and not the other one.

If a process is running that needs access to a resource on the

3 Project Inspiration

Moore's law is coming to an end (or so the internet says).

Consolidate all resources into one "system".

Sounded cool

4 Project Rough Plan

- Create a communication link between the two (or more) kernels
- Produce a distributed queue
- Change the process queue to this distributed queue instead. The kernel will probably only take own processes at this point.
- Change resource mappings so there are no conflicts (e.g. kernel A has eth0 and kernel B has eth1).
- Make it so if kernel A needs to access eth1, it uses the communication link
- Each kernel can take any *new* process from the queue. The process should stay on the same kernel at this point(so that memory doesn't need to be copied between machines)
- Allow an already started process to change kernel by copying the processes memory.
- Deal with kernel drop-out somehow?