Manipulating Jobs and Processes

Chris Green
DEVOPS ENGINEER

direct-root.com

Overview

Variables

- Appropriate scope

Process Priority

- Niceness

Signals

- Communicating with processes

Tactically suspending jobs

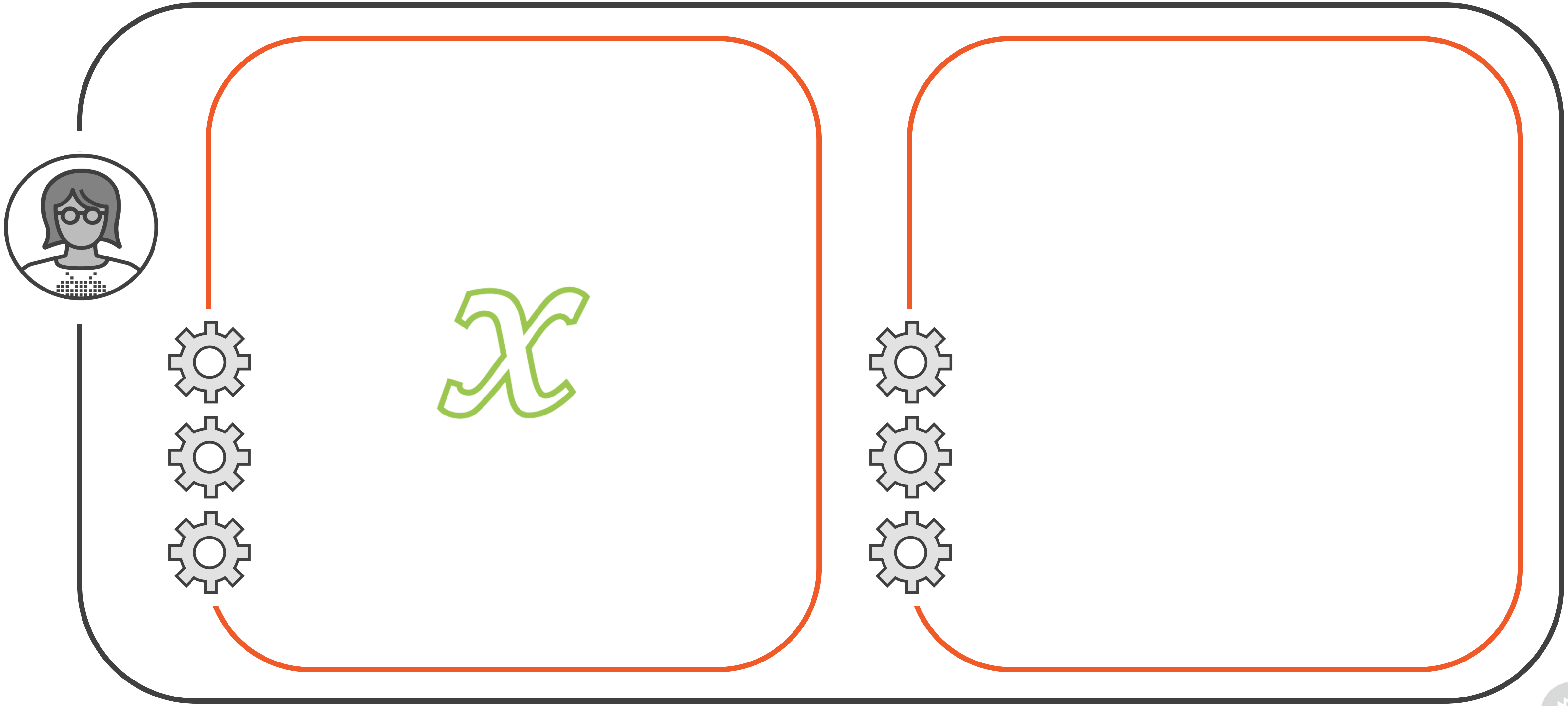
- Pause to think
- Free up CPU

Variables

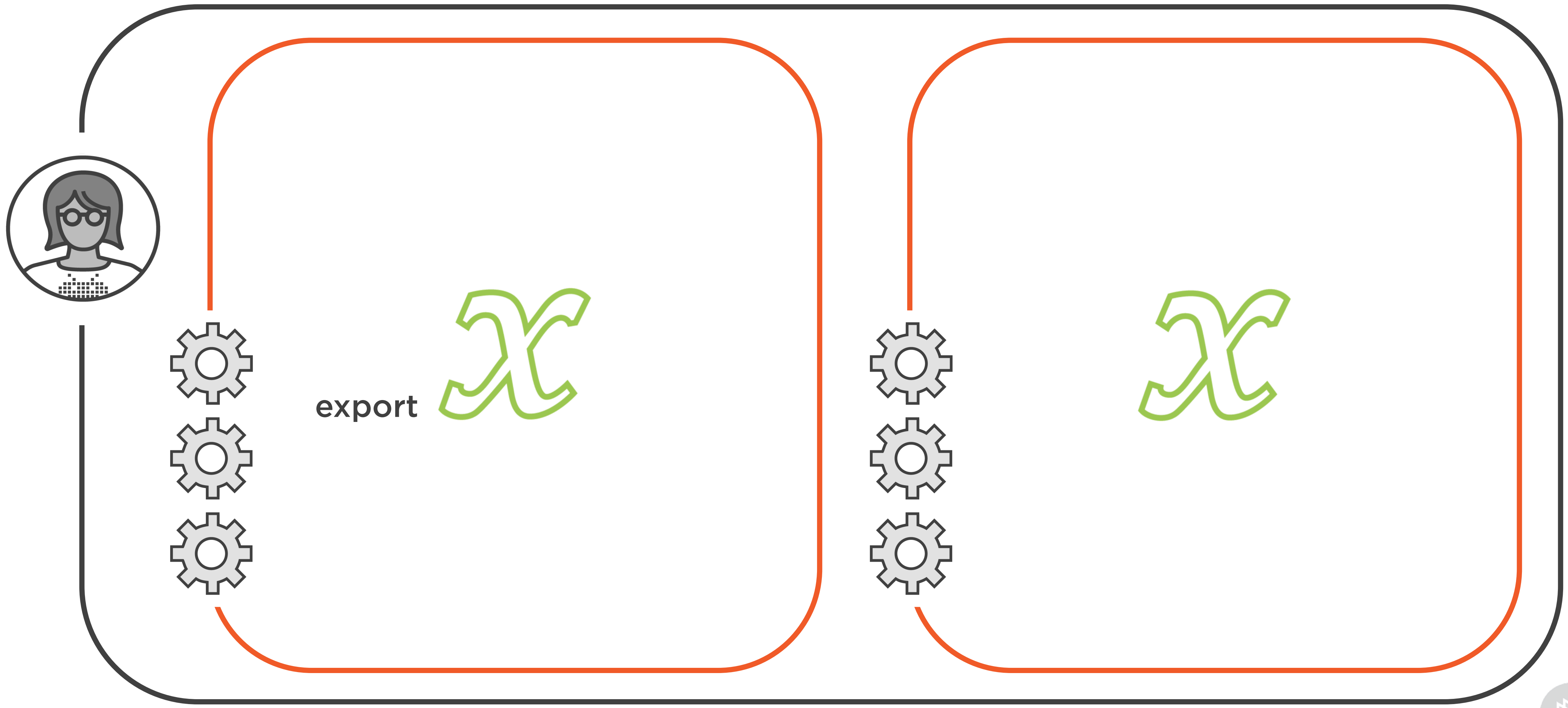
Local Versus Global

Local	Global
Standard assignment	Assigned with “export”
Available only in assigning job	Available in all spawned jobs and subprocesses
Resolve using dollar syntax	Resolve using dollar syntax

Local Scope



Global Scope



```
chris@ubuntu:~$ treat=cake
chris@ubuntu:~$ echo $treat
cake
chris@ubuntu:~$
```

Local Assignment

Use standard assignment, name=value

Resolve using $name

```
chris@ubuntu:~$ export treat=cake
chris@ubuntu:~$ echo $treat
cake
chris@ubuntu:~$
```

Global Assignment

Add an “export” to the assignment

Resolve using $name

```
chris@ubuntu:~$ export which_bash=$(which bash)
chris@ubuntu:~$ echo $which_bash
/bin/bash
chris@ubuntu:~$
```

Assign Command Output

Wrap a command in $()

Still resolve the variable with $name

Demo

Starting jobs

- With local variables
- With global variables

Process Priority

Scheduling

Limited resources, requires a running order

Higher priority runs first

Identical priorities run interchangeably

Nice number changes underlying priority

Priority (PRI)

Values -100 to 39

Higher number, yields to lower

Negative numbers considered “real time”

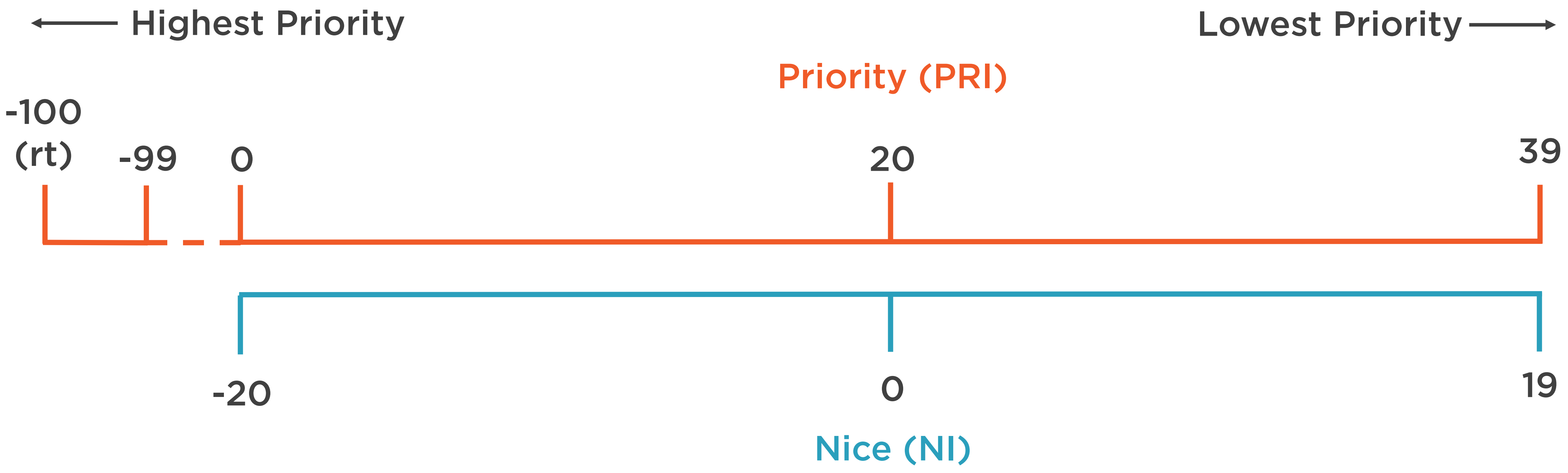
Play Nice!

Values -19 to 20

Higher number, yields to lower

Maps to underlying priority

Nice to Priority Mapping



Demo

Niceness

- nice
- htop

Signals

Signals

Inter-process communication

Asynchronous event notification

Signals have number, different between OS

Optional signal handlers

Most signals can be ignored

Related to interrupts, but different

Signals vs Interrupts

Signals	Interrupts
Can stop execution to take some action	Can stop execution to take some action
Communication from Linux kernel or process to another process	Communication from CPU to Linux kernel
Stop (suspend) a process, terminate a process, or dump memory	Handle divide by 0, page faults or accept hardware input

Common Signals: Hang Up

HUP

Sent to a background job, when the spawning foreground job is ended

Related “nohup” utility

Common Signals: Interrupt

INT

Ctrl+C

Commonly used to stop runaway foreground jobs

Common Signals: Quit

QUIT

Commonly used for terminating a process, whilst dumping its memory

Common Signals: Kill

KILL

Terminating stubborn processes, cannot be ignored

Common Signals: Terminate

TERM

Commonly issued by other software for terminating a process

Common Signals: Stop

STOP

Stop (suspend) a running process, cannot be ignored

Common Signals: Stp

STP

Ctrl+Z

Stop (suspend) a running process, can be ignored

Common Signals: Continue

CONT

Resume a stopped process

```
chris@ubuntu:~$ kill –s KILL 1234

chris@ubuntu:~$ kill –s STOP 5678

chris@ubuntu:~$ kill –s CONT 5678
```

Sending a Signal

Using the “kill” command

The –s flag allows specifying the signal name

Demo

Signals

Ending processes

- kill
- killall
- pkill
- htop
- xkill

Suspending Jobs Tactically

Suspending Jobs Tactically

Balancing act of interactive servers

Some jobs require more resources

Jobs being terminated causes frustration

You could try suspending when

- Disk is filling up
- CPU is needed
- More RAM will be available later

Demo

Stop a running process

See effect on the system