# Raspberry Pi Cluster

Para correr el codigo en un cluster de Raspberry Pis, seguimos estos pasos.

## Subiendo el firmware

Cada Raspberry Pi va a estar corriendo el mismo codigo y el mismo proyecto de nerves, el unico cambio es el hostname. En realidad no es esencial pero hace la configuracion mas facil. Para hacer esto, cambiamos el archivo `config/target.exs`, haciendo el siguiente cambio:

```elixir
  hosts: [:hostname, "nodeX"],
```

Donde `X` es el numero de nodo. Subimos el firmware a cada nodo con los siguientes comandos:

```bash
$ export MIX_TARGET=rpi4
$ mix deps.get
$ mix firmware
$ mix burn
```

Ahora podemos acceder al nodo facilmente con `ssh nodeX.local`.

## Conectando los raspis

Ahora, tenemos que conectar todos los nodos. En cada raspi, corremos:

```bash
iex> Toolshed.cmd("epmd -daemon")
iex> Node.start(:"nodeX@xxx.xxx.xxx.xxx")
iex> Node.set_cookie(:my_cookie)
```

Donde `nodeX` es el hostname del nodo y `xxx.xxx.xxx.xxx` es la direccion IP del nodo, que podemos encontrar con `Toolshed.ifconfig()`. Conectamos los raspis con:

```iex
iex> Node.connect(:"nodeX@xxx.xxx.xxx.xxx")
```

## Corriendo el codigo

Ahora con todos los nodos conectados, podemos correr el codigo en todos los nodos conectados, la distribucion de tareas se hace con

```elixir
nodes = Nodes.list
chunked = Enum.chunk_every(data, length: div(length(data), length(nodes)))
Enum.zip(nodes, chunked)
|> Enum.map(fn {node, chunk} ->
 Node.spawn(node, MyModule, function, [chunk])
end)
|> Enum.map(fn -> receive do x -> x end end)
|> Enum.reduce(&MyModule.collect_results/2)
```

Como todos los nodos tienen el mismo codigo, no hay problema al correr `Node.spawn/4`. Asi seria el esquema para distribuir la tarea entre todos los nodos conectados.