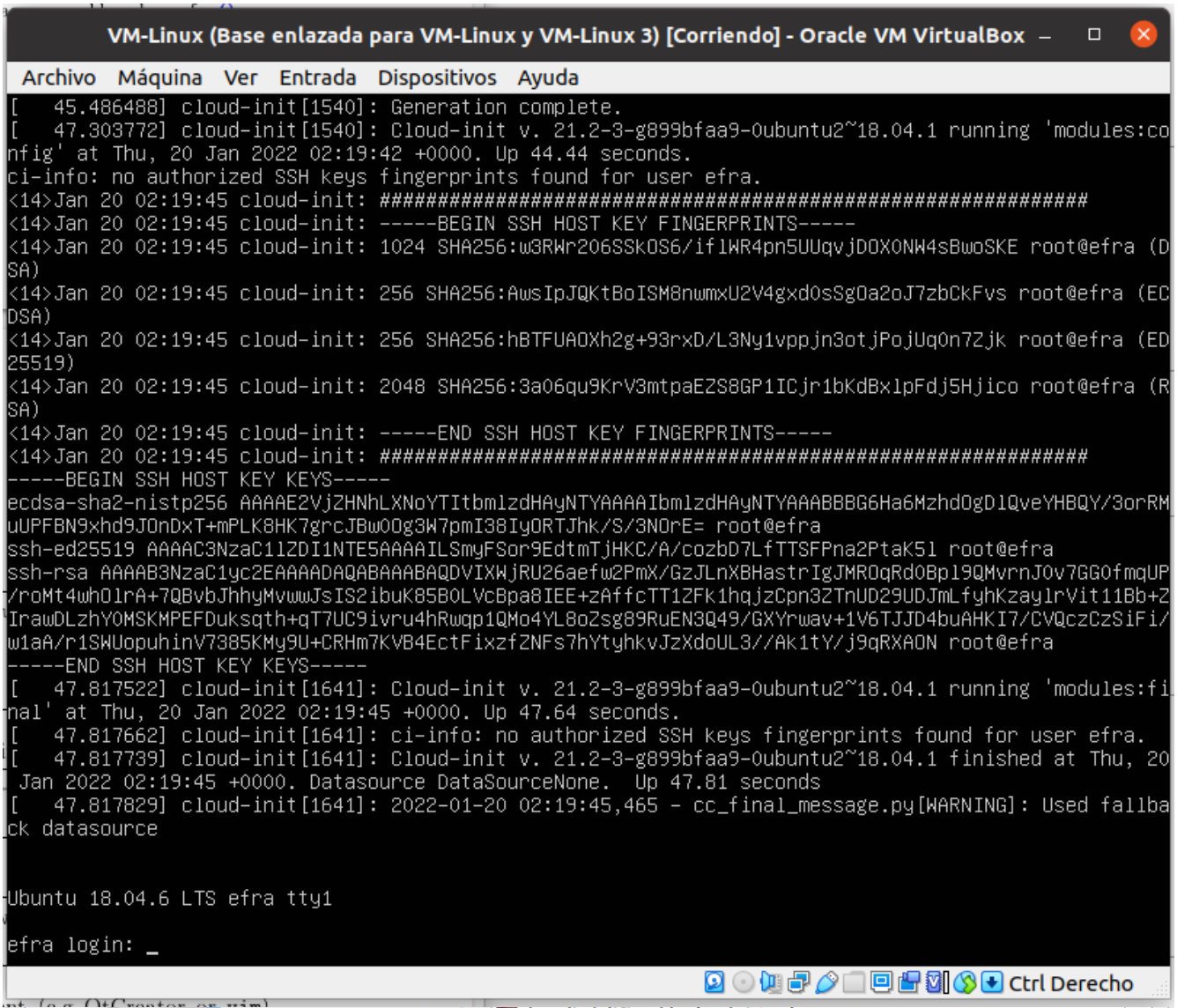
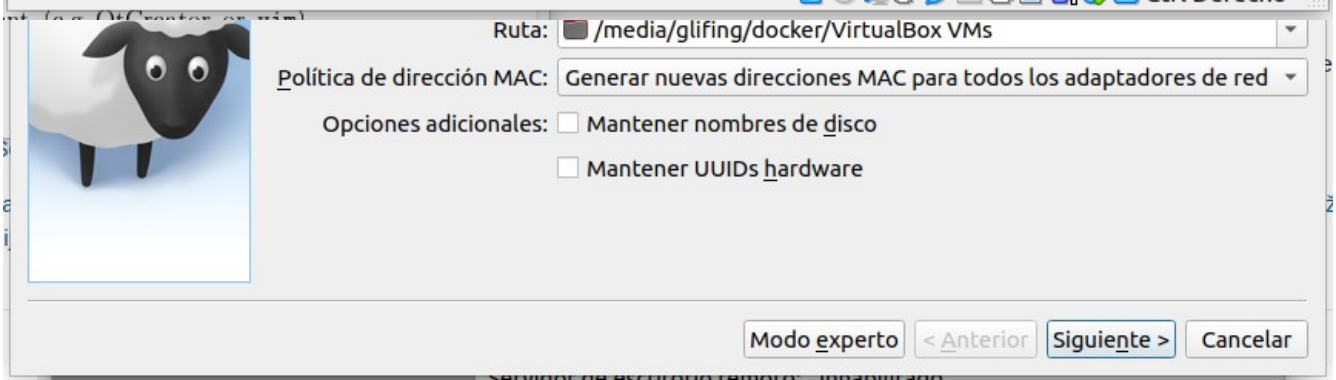
Cloud Computing and Deployment

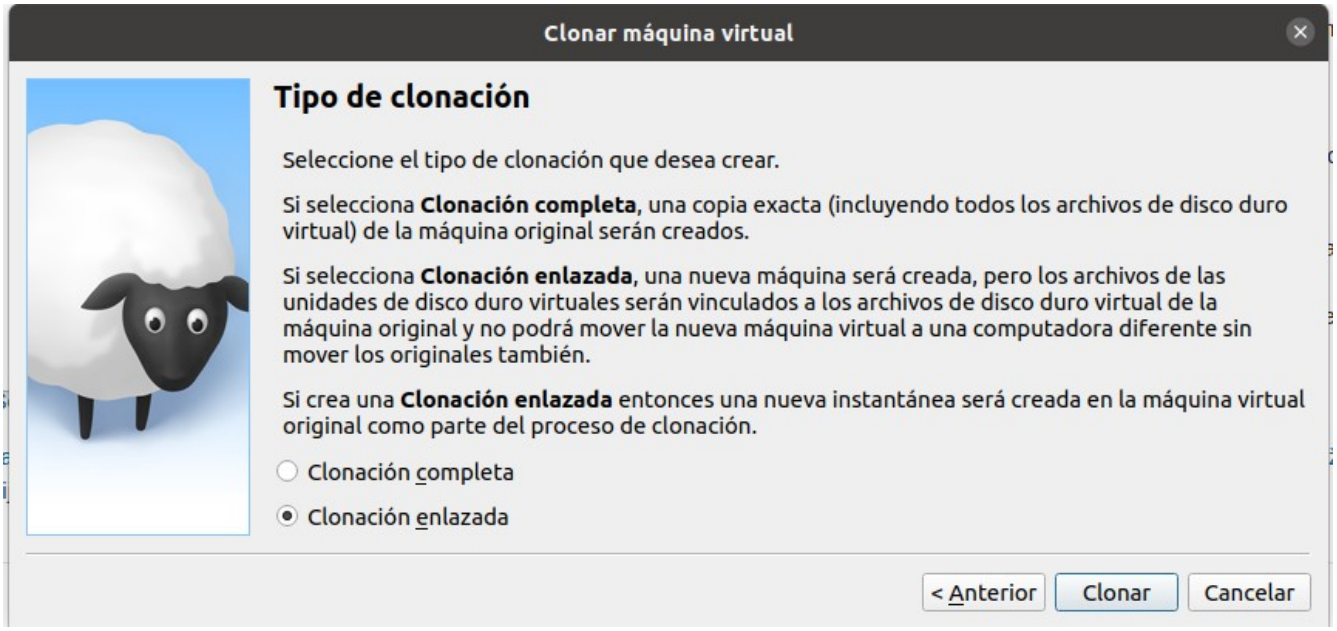
Efraín Fernández Sangrador

3. N03-MPI-PingPong

I lost all my work because I was working with an Ubuntu Partition, so I return to install Ubuntu server with ssh and for save memory, I am going to clone one VM with an linked clone and new MACs for have a three-machines cluster.







For do this ping pong and for the cluster, we can link the machines with a new network connected by the interface 1 of each machine, and if the DHCP doesn't work, change manually the IPs (I know this is not the right way, but I am having a lot of problems with my laptop).

We can install the OpenMPI communication interface with the command:

sudo -S apt-get install -y \ libopenmpi-dev openmpi-bin g++;

If all will be okey, we could do this in all the cluster with ssh.

```
$ stty -echo; echo "Password: "; read pass; stty echo
  ↪ ;
$ for i in {1..10}; do \
   echo $pass | ssh -p 200$i 127.0.0.1 \
     sudo -S apt-get install -y \
       libopenmpi-dev openmpi-bin g++;
  done
```

OpenMPI is a Message Passing Interface for wok with distributed memory. This is divided into several processes and each node is driven by one (or more) processes. The language used is mpicc, closed to the languaje C. We can add this code by nano as N03-MPI-PingPong.c:

```
#include <mpi.h>

#include <stdlib.h>

#include <stdio.h>
```

```
void rank0() {
	MPI_Status status;
	int nPingPongs = 0;
	float sum = 0;

	while (1) {
		for (int i = 1; i < 10; i++) {
			float pongReceive, pingSend = rand()%18000 / 100.0;
			MPI_Send(&pingSend, 1, MPI_FLOAT, i, 42, MPI_COMM_WORLD);
			MPI_Recv(&pongReceive, 1, MPI_FLOAT, i, 42, MPI_COMM_WORLD,
			&status);
			nPingPongs++;
			sum += pongReceive;
			if (sum > 360) sum -= 360;
			if (sum >= 270.505 && sum <= 270.515) {
				printf("Number of Ping-Pongs: %d\n", nPingPongs);
				pingSend = -42; // exit ping pong
				for (int i = 1; i < 10; i++)
				MPI_Send(&pingSend, 1, MPI_FLOAT, i, 42, MPI_COMM_WORLD);
				return;
			}
		}
	}
}
void rankN(int N) {
	MPI_Status status;
	while (1) {
		float pingpong42;
		MPI_Recv(&pingpong42, 1, MPI_FLOAT, 0, 42, MPI_COMM_WORLD, &status);
```

```
            if (pingpong42 == -42) return;
            MPI_Send(&pingpong42, 1, MPI_FLOAT, 0, 42, MPI_COMM_WORLD);
        }
}

int main(int argc, char* argv[]) {
        int rank;
        MPI_Init(&argc, &argv);
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);
        if (!rank) rank0(); else rankN(rank);
        MPI_Finalize();
}
```

Functions used: (doc: https://www.open-mpi.org/doc/v3.0/)

MPI_Send

```
int MPI_Send(const void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)
```

Input Parameters

buf
: Initial address of send buffer (choice).

count
: Number of elements send (nonnegative integer).

datatype
: Datatype of each send buffer element (handle).

dest
: Rank of destination (integer).

tag
: Message tag (integer).

comm
: Communicator (handle).

Output Parameter

IERROR
: Fortran only: Error status (integer).

MPI_Recv

```
#include <mpi.h>
int MPI_Recv(void *buf, int count, MPI_Datatype datatype,
     int source, int tag, MPI_Comm comm, MPI_Status *status)
```

Input Parameters

count
: Maximum number of elements to receive (integer).

datatype
: Datatype of each receive buffer entry (handle).

source
: Rank of source (integer).

tag
: Message tag (integer).

comm
: Communicator (handle).

Output Parameters

buf
: Initial address of receive buffer (choice).

status
: Status object (status).

IERROR
: Fortran only: Error status (integer).

In this point, we only have to copy this file to each node (via ssh or manually)

```
$ for i in {1..10}; do
        vboxmanage controlvm N03−MPI−PingPong−$i \
        nic1 hostonly vboxnet1;
done
```

And compile it in each node:

```
$ parallel−ssh −i \
        $(echo "-H " 192.168.57.{3..12}) \
        mpicc N03−MPI−PingPong.c \
        −o N03−MPI−PingPong
```

For execute

```
$ time ssh 192.168.57.3 \
        mpiexec −n 10 \
                $(echo "-H " 192.168.57.{3..12}) \
                −−mca btl\_base\_warn\_component\_unused 0 \
                N03−MPI−PingPong
```

This would be the result
Number of Ping−Pongs: 7592
real 1m16,476s
user 0m0,039s
sys 0m0,018s