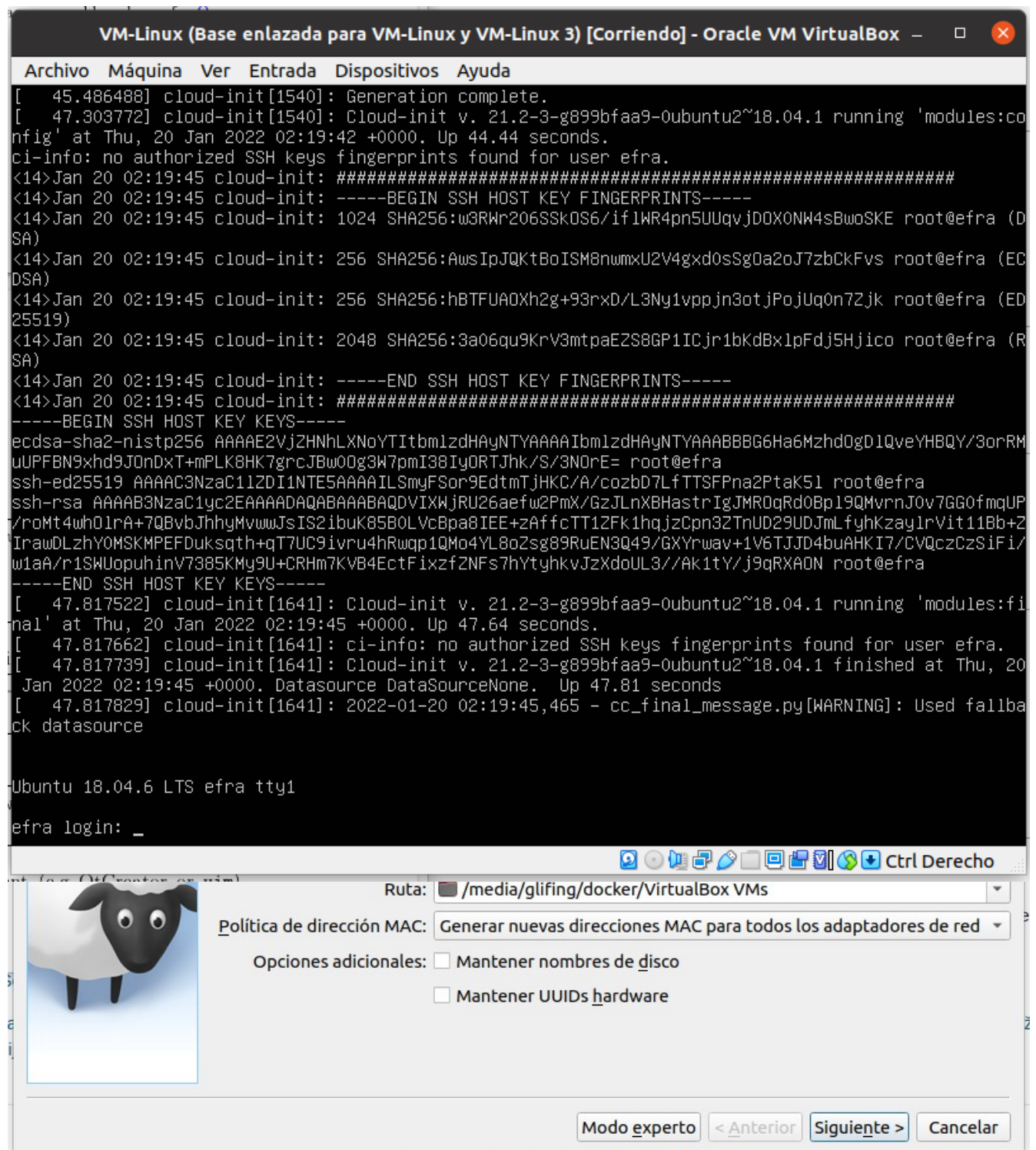


# Cloud Computing and Deployment

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.

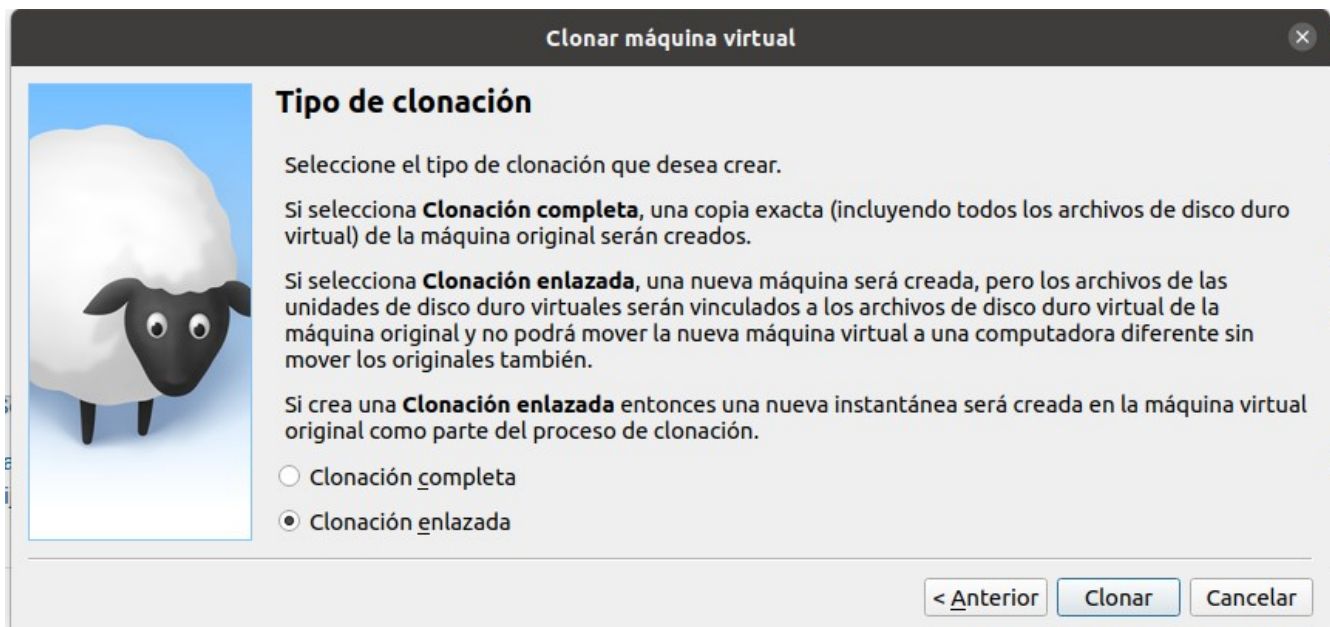


The screenshot displays the Oracle VM VirtualBox interface. The top window, titled "VM-Linux (Base enlazada para VM-Linux y VM-Linux 3) [Corriendo] - Oracle VM VirtualBox", shows a terminal output from a cloud-init process. The output indicates that the cloud-init v. 21.2-3-g899bfaa9-0ubuntu2~18.04.1 is running successfully, generating SSH host key fingerprints and keys. The terminal shows the following output:

```
[ 45.486488] cloud-init[1540]: Generation complete.
[ 47.303772] cloud-init[1540]: Cloud-init v. 21.2-3-g899bfaa9-0ubuntu2~18.04.1 running 'modules:config' at Thu, 20 Jan 2022 02:19:42 +0000. Up 44.44 seconds.
ci-info: no authorized SSH keys fingerprints found for user efra.
<14>Jan 20 02:19:45 cloud-init: #####
<14>Jan 20 02:19:45 cloud-init: -----BEGIN SSH HOST KEY FINGERPRINTS-----
<14>Jan 20 02:19:45 cloud-init: 1024 SHA256:w3RWr206SSk0S6/1f1WR4pn5UUqvjD0X0NW4sBwoSKE root@efra (DSA)
<14>Jan 20 02:19:45 cloud-init: 256 SHA256:AwsIpJQKtBoISM8nmwU2V4gxd0sSg0a2oJ7zbCkFvs root@efra (ECDSA)
<14>Jan 20 02:19:45 cloud-init: 256 SHA256:hBTfUA0Xh2g+93rxD/L3Ny1vppjn3otjPojUq0n7Zjk root@efra (ED25519)
<14>Jan 20 02:19:45 cloud-init: 2048 SHA256:3a06qu9KrV3mtpaEZS8GP1ICjr1bKdBx1pFdj5Hjico root@efra (RSA)
<14>Jan 20 02:19:45 cloud-init: -----END SSH HOST KEY FINGERPRINTS-----
<14>Jan 20 02:19:45 cloud-init: #####
-----BEGIN SSH HOST KEY KEYS-----
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBG6Ha6Mzhd0gD1QveYHBQY/3orRM
uUPFB9xhd9J0nDxt+mPLK8HK7grcJBw00g3W7pmI38IyORTJhk/S/3N0rE= root@efra
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAILSmYFSor9EdtmTjHKC/A/cozbd7LfTTSFPna2PtaK5l root@efra
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQADVIxWjRU26aefw2PmX/GzJLnXBHastrIgJMR0gRd0Bp19QMvrnJ0v7GG0fmqUP
/roMt4wh01ra+7QBvbJhhyMvwuJsIS2ibuK85B0LVcBpa8IEE+zAffcTT12Fk1hqjzCpn32TnUD29UDJmLfyhKzaylrVit11Bb+Z
IrawDLzhY0MSKMPEDuksqth+qT7UC9ivru4hRwqp1QMo4YL8o2sg89RUEN3Q49/GXYrwav+1V6TJJD4buAHKI7/CVQczCzSiFi/
wlaA/r1SWUopuhinV7385KM9U+CRHm7KVB4EctFixzf2NFS7hYtyhkvJzXdoUL3//Ak1tY/j9qRXA0N root@efra
-----END SSH HOST KEY KEYS-----
[ 47.817522] cloud-init[1641]: Cloud-init v. 21.2-3-g899bfaa9-0ubuntu2~18.04.1 running 'modules:final' at Thu, 20 Jan 2022 02:19:45 +0000. Up 47.64 seconds.
[ 47.817662] cloud-init[1641]: ci-info: no authorized SSH keys fingerprints found for user efra.
[ 47.817739] cloud-init[1641]: Cloud-init v. 21.2-3-g899bfaa9-0ubuntu2~18.04.1 finished at Thu, 20 Jan 2022 02:19:45 +0000. Datasource DataSourceNone. Up 47.81 seconds
[ 47.817829] cloud-init[1641]: 2022-01-20 02:19:45,465 - cc_final_message.py[WARNING]: Used fallback
ck datasource

Ubuntu 18.04.6 LTS efra tty1
efra login: _
```

The bottom window shows the "Settings" dialog for the VM. The "Ruta" field is set to "/media/glifing/docker/VirtualBox VMs". The "Política de dirección MAC" is set to "Generar nuevas direcciones MAC para todos los adaptadores de red". The "Opciones adicionales" section has two checkboxes: "Mantener nombres de disco" (unchecked) and "Mantener UUIDs hardware" (unchecked). The "Modo experto" button is selected, and the "Anterior" and "Siguiente >" buttons are visible.



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 okay, 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 work 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 language 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