*****Instituto Politécnico Nacional***

***Escuela Superior de Cómputo***

*Desarrollo de Sistemas Distribuidos*

***Tarea 4. Implementación de un token-ring***

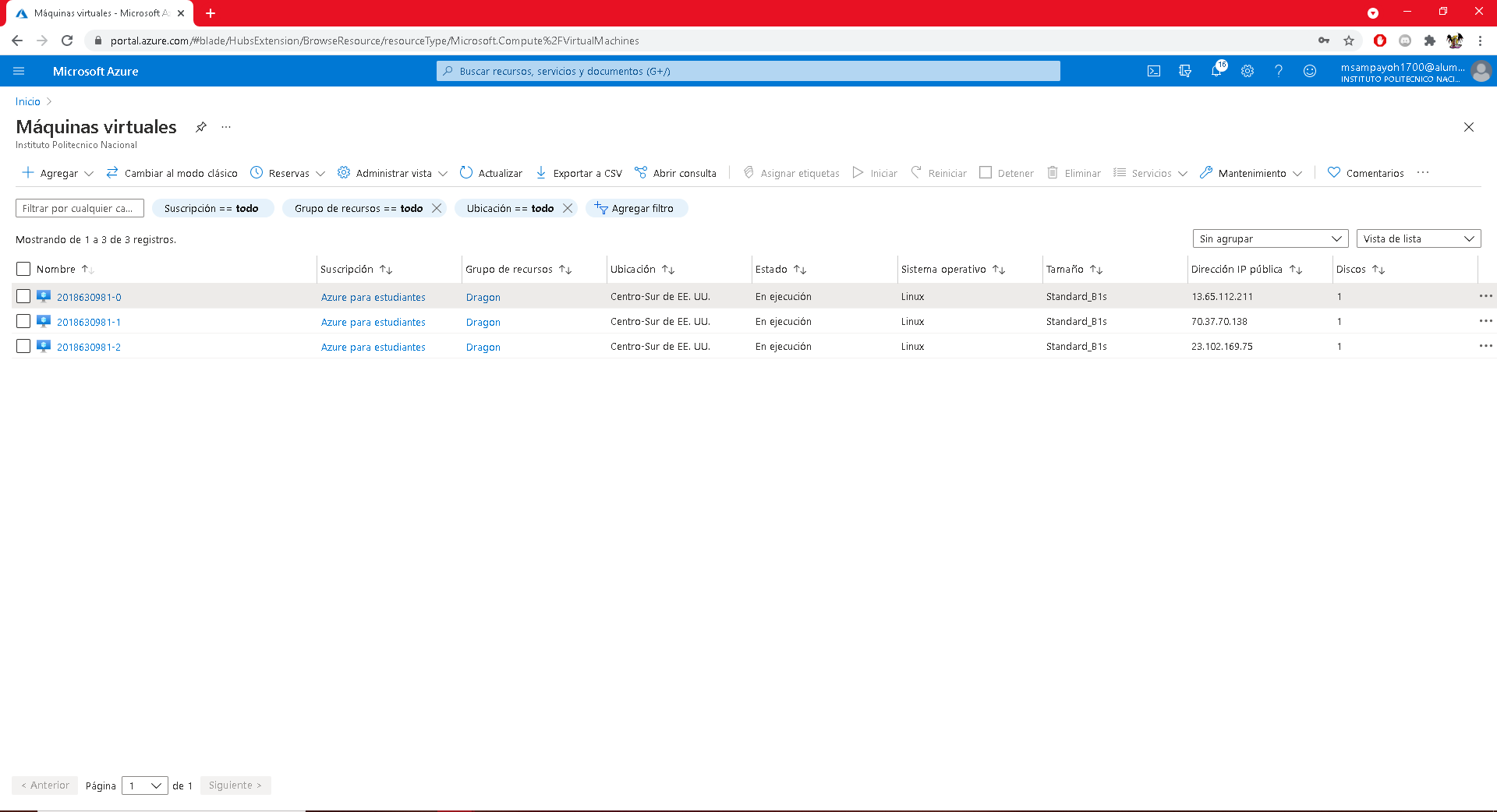
***Nombre:*** *Sampayo Hernández Mauro*

***Grupo:*** *4CV1*

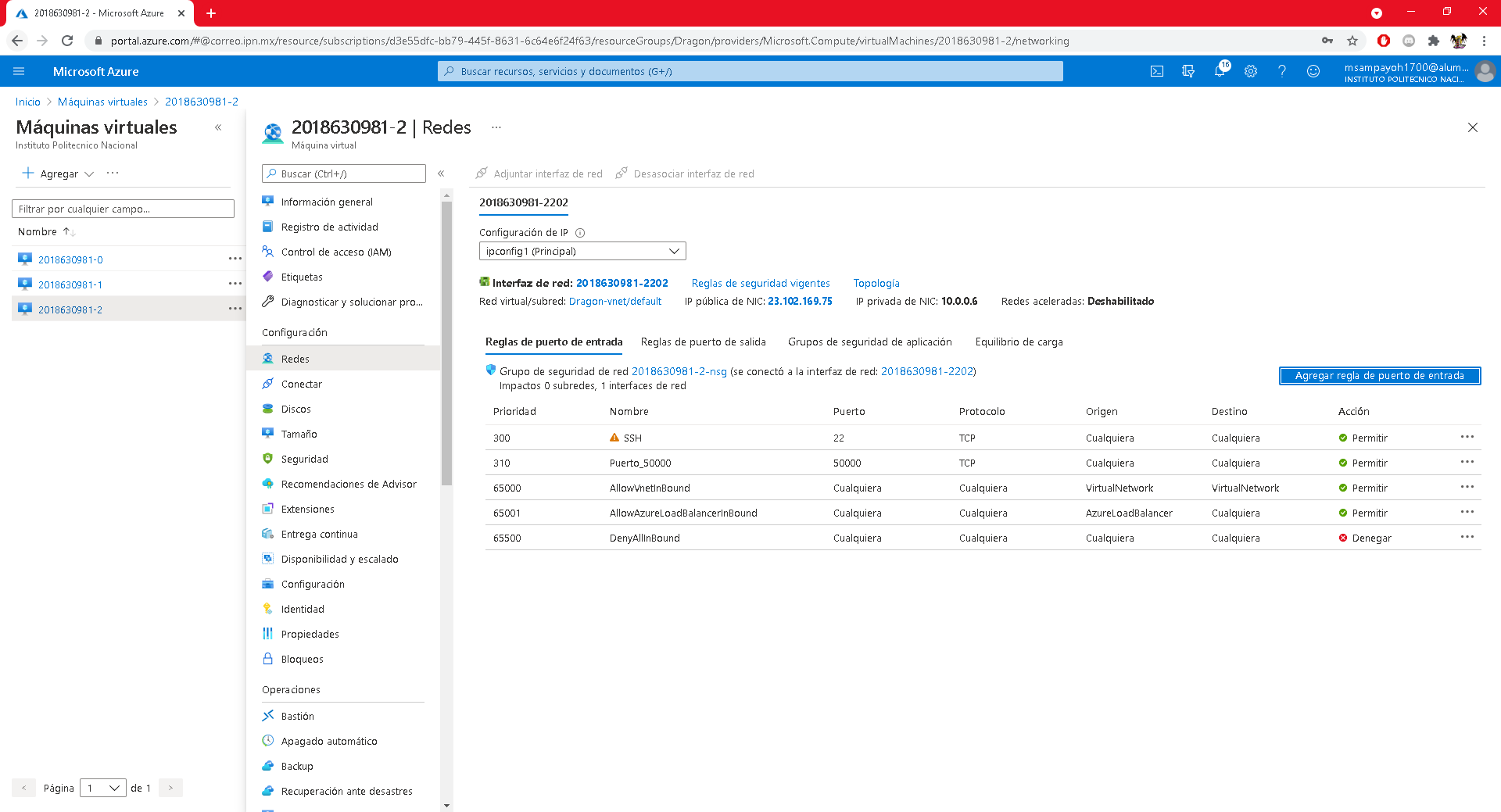
***Profesor:*** *Pineda Guerrero Carlos*

**Creación de las máquinas virtuales:**

Primero se realiza la creación de 3 máquinas virtuales de Ubuntu en Azure, cada una con 1 CPU, 1 GB de RAM y disco HDD estándar.

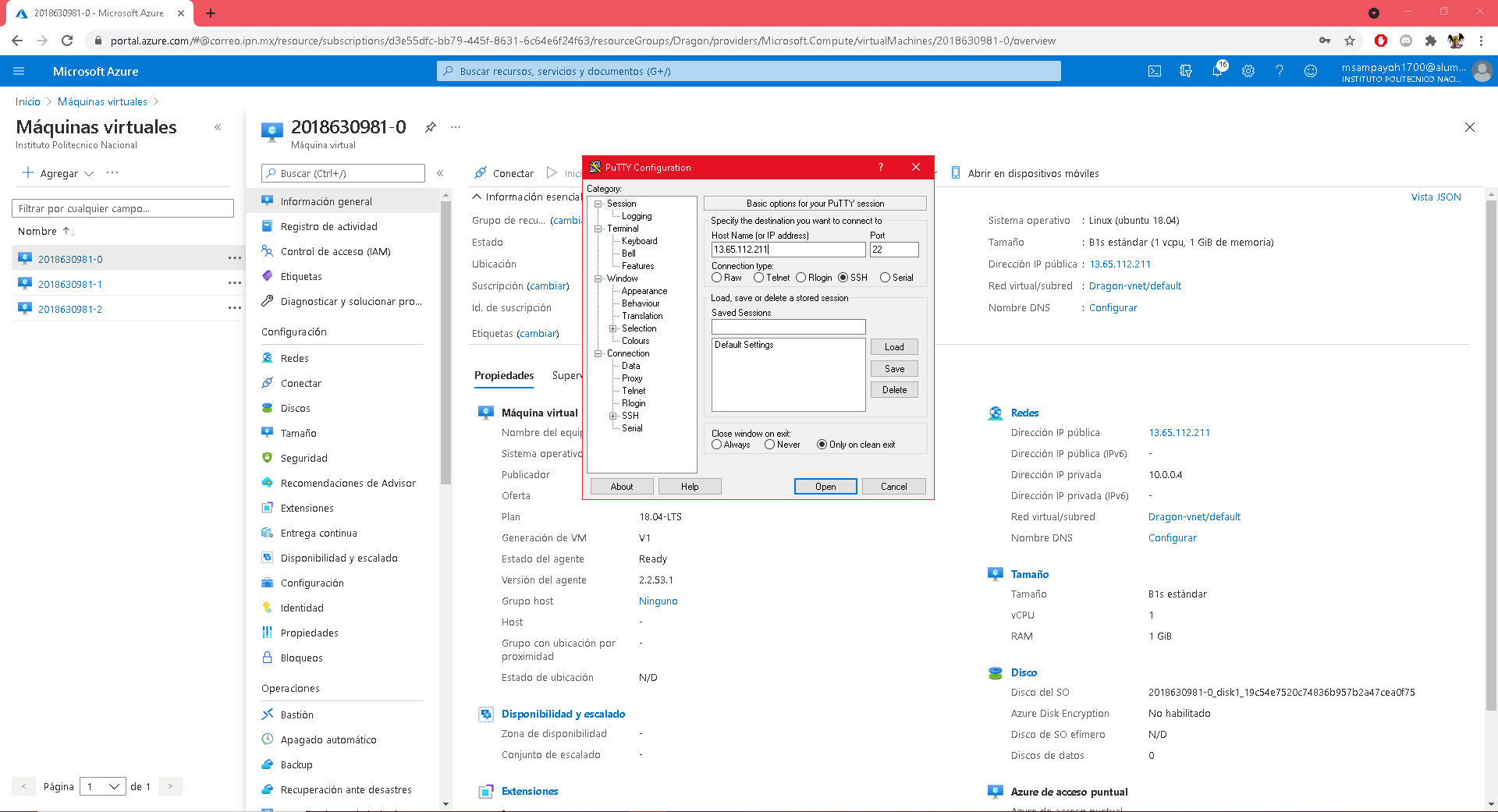


Configuramos cada máquina virtual para que puedan recibir conexiones a través del puerto 50000, abriendo dicho puerto en cada máquina virtual y seleccionando el protocolo de TCP para dicho puerto. A continuación, se muestra dicho puerto configurado en la máquina virtual del nodo dos.

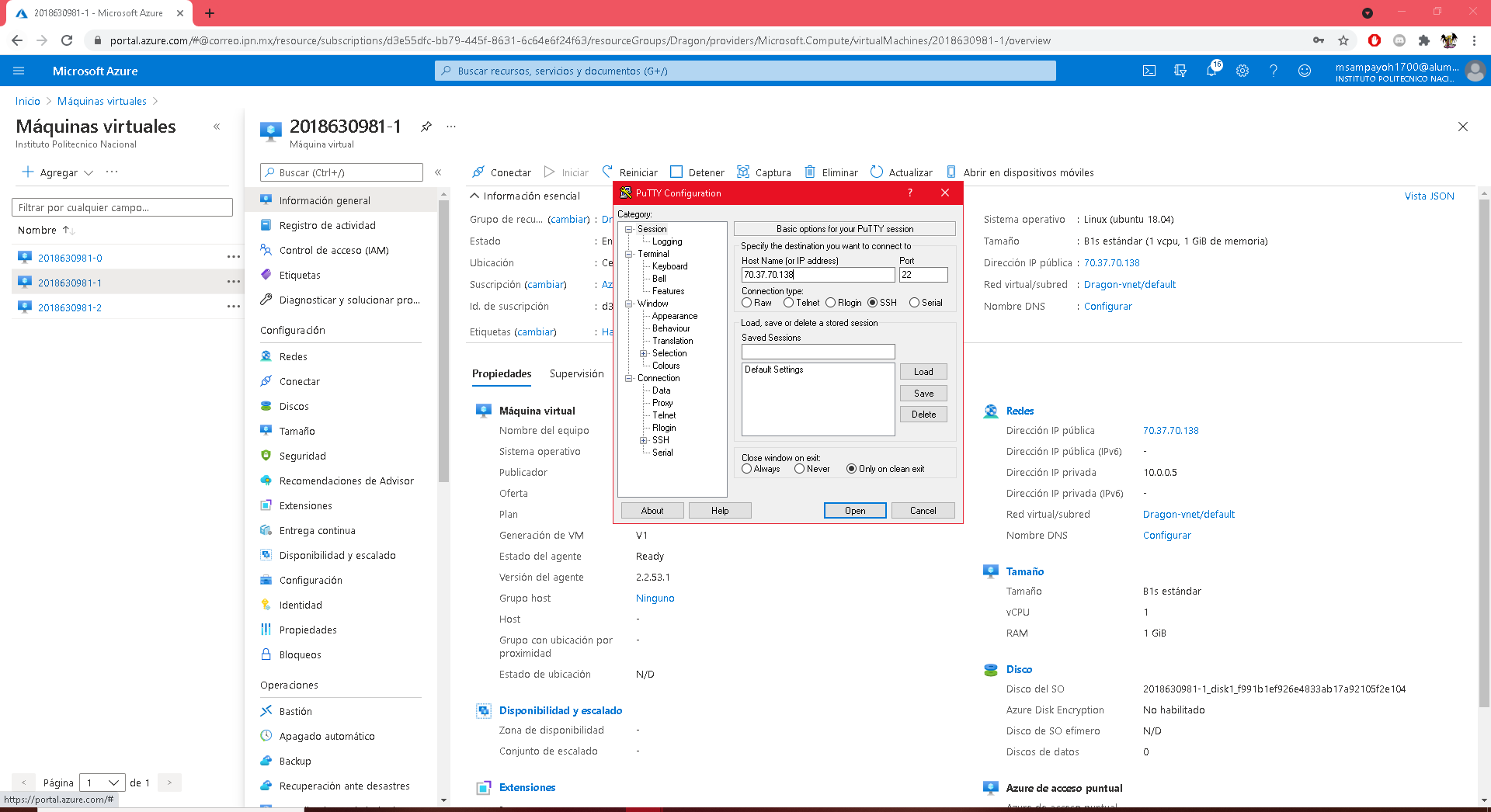


Ejecutamos Putty.exe, para así acceder a cada una de las máquinas virtuales por medio de sus respectivos IPs.

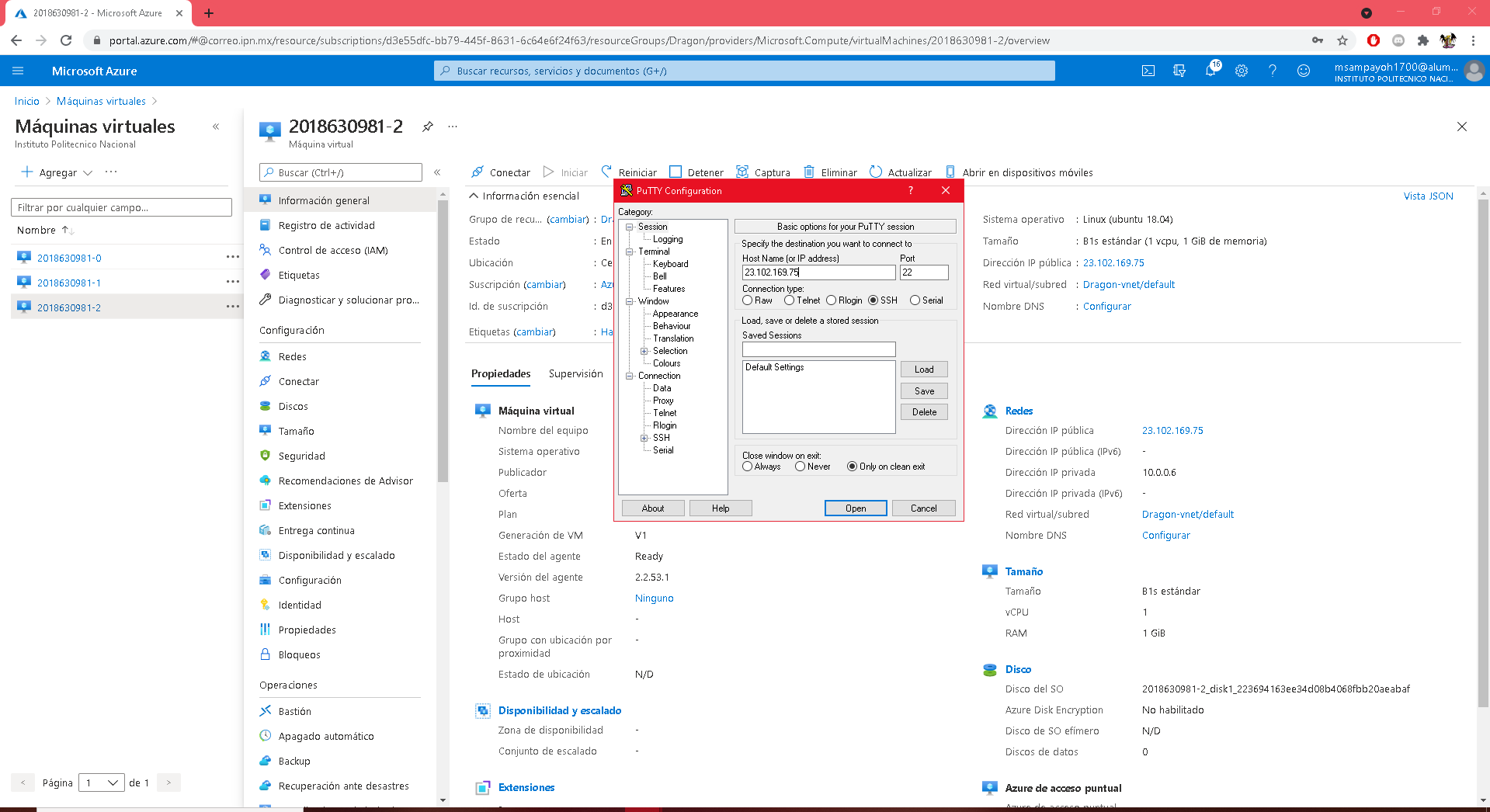
Aquí se muestra el acceso a la máquina virtual para el nodo 0:



Aquí se muestra el acceso a la máquina virtual para el nodo 1:

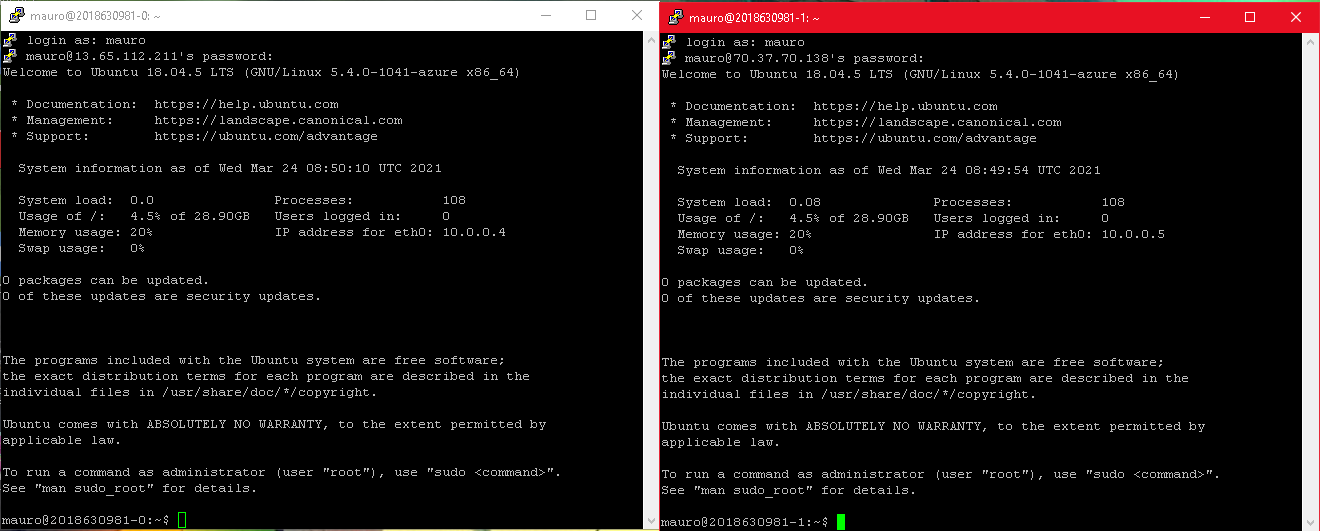


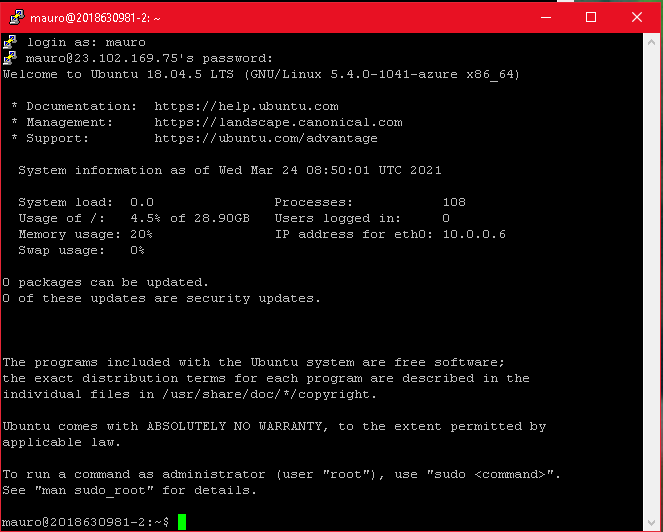
Finalmente se muestra el acceso a la máquina virtual para el nodo 2:



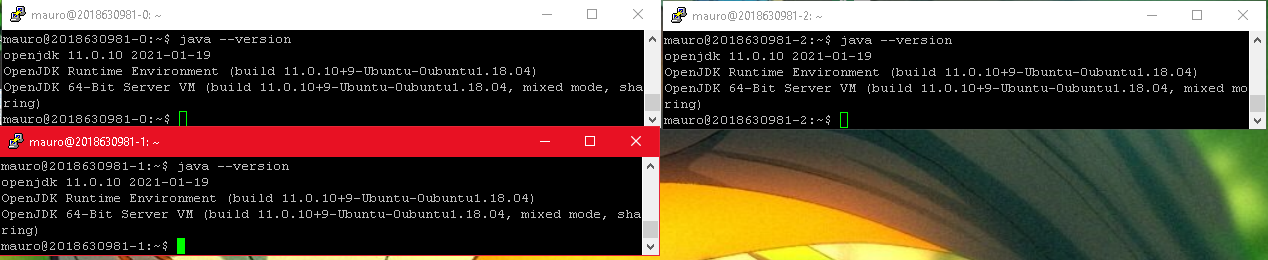
**Compilación del programa:**

Una vez iniciamos cada una de las máquinas virtuales, accedemos a ellas por medio del usuario y la contraseña que configuramos previamente al momento de realizar la creación de las mismas

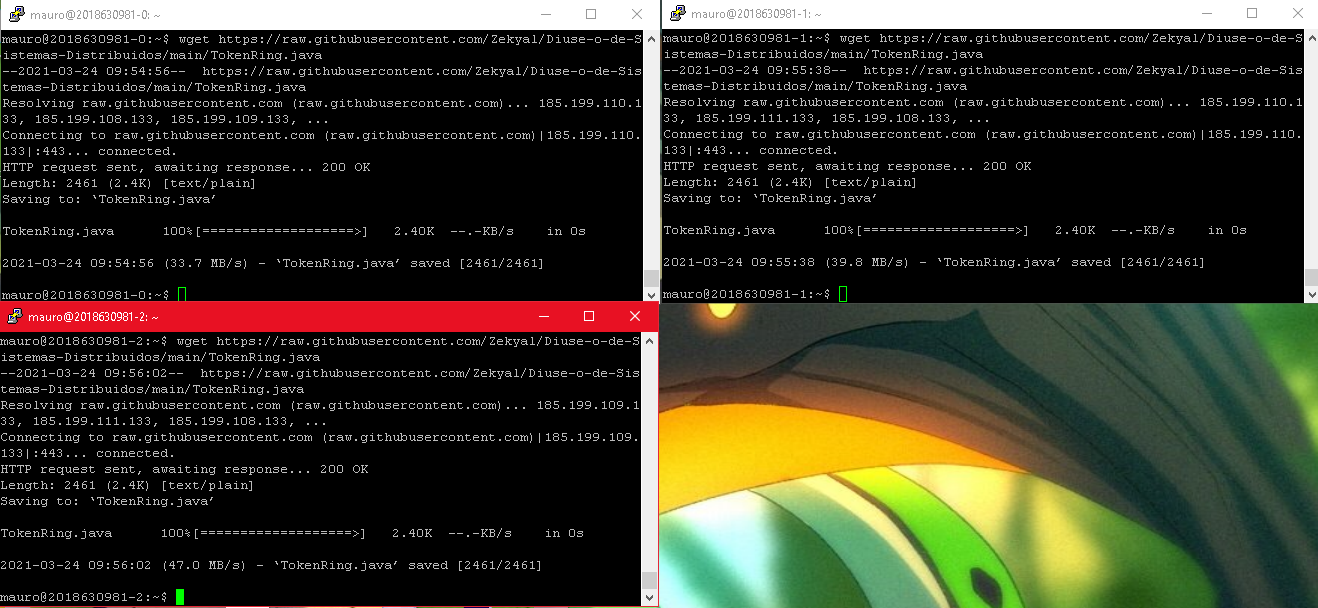




Instalamos el jdk 8 para todas las maquinas virtuales por medio del comando “sudo apt install openjdk-8-jdk”, y comprobamos que la instalación se haya realizado correctamente por medio del comando “java --version”



Obtenemos el archivo TokenRing.java, el cuál fue previamente subido en un repositorio de github por medio del comando “wget <https://raw.githubusercontent.com/Zekyal/Diuse-o-de-Sistemas-Distribuidos/main/TokenRing.java>” para realizar su compilación. Cabe destacar que el link utilizado en el comando “wget” nos lleva a la versión raw del archivo que se encuentra en el repositorio.

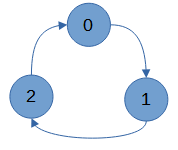


Una vez cargado el archivo dentro de cada máquina virtual, realizamos la compilación de dicho archivo por medio del comando “javac TokenRing.java”.



**Ejecución del programa:**

Realizamos la ejecución del programa pasándole 2 argumentos al programa, el primero que indicará el número de nodo y el segundo que indicará la ip de la máquina virtual que se usará como el nodo siguiente, de tal manera que la configuración del token ring quede de la siguiente manera:

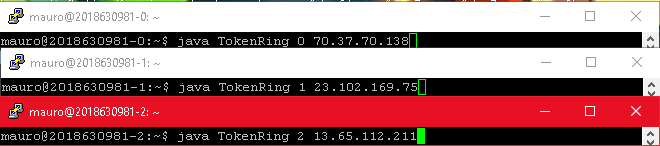


El token a enviar será un un número entero de 32 bits, el cual será inicializado en el nodo 0 inicializará con el valor de1. Posteriormente el nodo 0 enviará el token al nodo 1, entonces el nodo 1 recibirá el token y lo enviará al nodo 2. El nodo 2 recibirá el token y lo enviará al nodo 0.

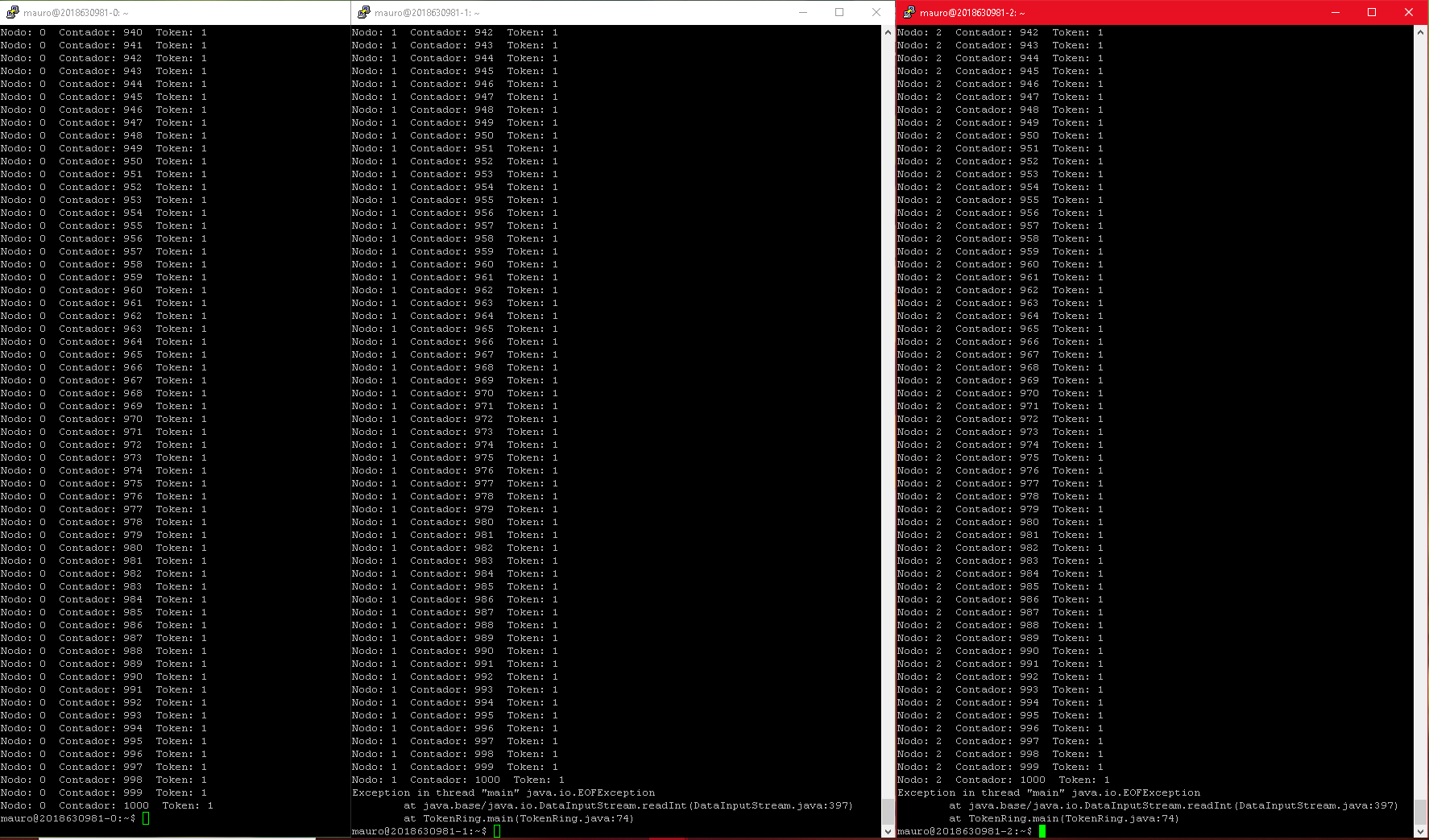
Las direcciones IP de cada nodo se enlistan a continuación:

* Dirección IP del nodo 0: 13.65.112.211
* Dirección IP del nodo 1: 70.37.70.138
* Dirección IP del nodo 2: 23.102.169.75

Se muestra a continuación la ejecución del programa en cada una de las máquinas virtuales:



Finalmente se muestra la salida de los 3 nodos una vez el nodo llega al valor de 1000 en su contador. Como se puede observar, al haber finalizado el nodo 0, y por consiguiente no haber ningún token a enviar para los nodos 1 y 2, estos generan un error.



Finalmente, cade mencionar que el programa no realizara ninguna acción si es que los nodos no están conectados entre sí.

**Código Fuente del programa:**

**import** java**.**io**.**DataInputStream**;**

**import** java**.**io**.**DataOutputStream**;**

**import** java**.**net**.**ServerSocket**;**

**import** java**.**net**.**Socket**;**

public class TokenRing **{**

static DataInputStream entrada**;**

static DataOutputStream salida**;**

static boolean primera\_vez **=** **true;**

static String ip**;**

static int nodo**;**

static int token**;**

static int contador **=** 0**;**

static class Worker **extends** Thread

**{**

public void run**()**

**{**

//Algoritmo 1

**try{**

ServerSocket servidor **=** **new** ServerSocket**(**50000**);**

Socket conexion **=** servidor**.**accept**();**

entrada **=** **new** DataInputStream**(**conexion**.**getInputStream**());**

**}catch(**Exception e**){**

e**.**printStackTrace**();**

**}**

**}**

**}**

public static void main**(**String**[]** args**)** **throws** Exception

**{**

**if** **(**args**.**length **!=** 2**)**

**{**

System**.**err**.**println**(**"Se debe pasar como parametros el numero del nodo y la IP del siguiente nodo"**);**

System**.**exit**(**1**);**

**}**

nodo **=** Integer**.**valueOf**(**args**[**0**]);** // el primer parametro es el numero de nodo

ip **=** args**[**1**];** // el segundo parametro es la IP del siguiente nodo en el anillo

//Algoritmo 2

Worker w **=** **new** Worker**();**

w**.**start**();**

Socket conexion **=** **null;**

**for(;;)**

**try**

**{**

conexion **=** **new** Socket**(**ip**,**50000**);**

**break;**

**}**

**catch** **(**Exception e**)**

**{**

Thread**.**sleep**(**100**);**

**}**

salida **=** **new** DataOutputStream**(**conexion**.**getOutputStream**());**

w**.**join**();**

**for(;;){**

**if(**nodo**==**0**){**

**if(**primera\_vez**==true){**

primera\_vez **=** **false;**

token **=** 1**;**

**}else{**

token **=** entrada**.**readInt**();**

contador**++;**

System**.**out**.**println**(**"Nodo: "**+**nodo**+**" Contador: "**+**contador**+**" Token: "**+**token**);**

**if(**contador**==**1000**)**

**break;**

**}**

**}else{**

token **=** entrada**.**readInt**();**

contador**++;**

System**.**out**.**println**(**"Nodo: "**+**nodo**+**" Contador: "**+**contador**+**" Token: "**+**token**);**

**}**

salida**.**writeInt**(**token**);**

**}**

**}**

**}**

**Conclusión:**

El uso del algoritmo de token en anillo o token ring resulta bastante efectivo para llevar a cabo la exclusión mutua, pues además de que su implementación resulta ser bastante sencilla, es bastante rápida y ocupa pocos recursos siempre y cuando no se tenga un gran número de nodos y la conexión entre estos sea estable.