



Instituto Politécnico Nacional Escuela Superior de Cómputo

Tarea 6. Multiplicación de matrices utilizando objetos distribuidos

PRESENTA

Vladimir Azpeitia Hernández

PROFESOR

Carlos Pineda Guerrero

ASGINATURA

Desarrollo de sistemas distribuidos

20 de abril de 2021

Tarea 6. Multiplicación de matrices utilizando objetos distribuidos

Vladimir Azpeitia Hernández

20 de abril de 2021

1. Introducción

Cada alumno deberá desarrollar un sistema que calcule el producto de dos matrices cuadradas utilizando Java RMI, tal como se explicó en clase.

Se deberá ejecutar dos casos:

- $N=8$, se deberá desplegar las matrices A, B y C y el checksum de la matriz C.
- $N=1000$, deberá desplegar el checksum de la matriz C.

Los elementos de las matrices A, B y C serán de tipo float y el checksum será de tipo double.

Se deberá inicializar las matrices A y B de la siguiente manera (notar que la inicialización es diferente a la que se realizó en la tarea 3):

$$A[i][j] = i + 3 * j$$

$$B[i][j] = i - 3 * j$$

El servidor RMI ejecutará en dos máquinas virtuales (nodo 1 y nodo 2) con **Ubuntu** en Azure. El programa rmiregistry ejecutará en cada nodo donde ejecute el servidor RMI. El nodo 1 calculará los productos C1 y C2 mientras que el nodo 2 calculará los productos C3 y C4.

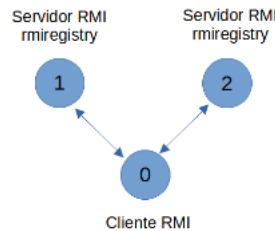


Figura 1: Topología del sistema

El cliente RMI ejecutará en una tercera máquina virtual con **Ubuntu** (nodo 0). El cliente RMI inicializará las matrices A y B, obtendrá la transpuesta de la matriz B, invocará el método remoto `multiplica_matrices()`, calculará el checksum de la matriz C, y en su caso ($N=8$) desplegará las matrices A, B y C.

Se deberá utilizar las funciones que vimos en clase: `separa_matriz()`, `multiplica_matrices()` y `acomoda_matriz()`.

2. Desarrollo

2.1. Instalación y configuración de la máquina virtual del nodo 0

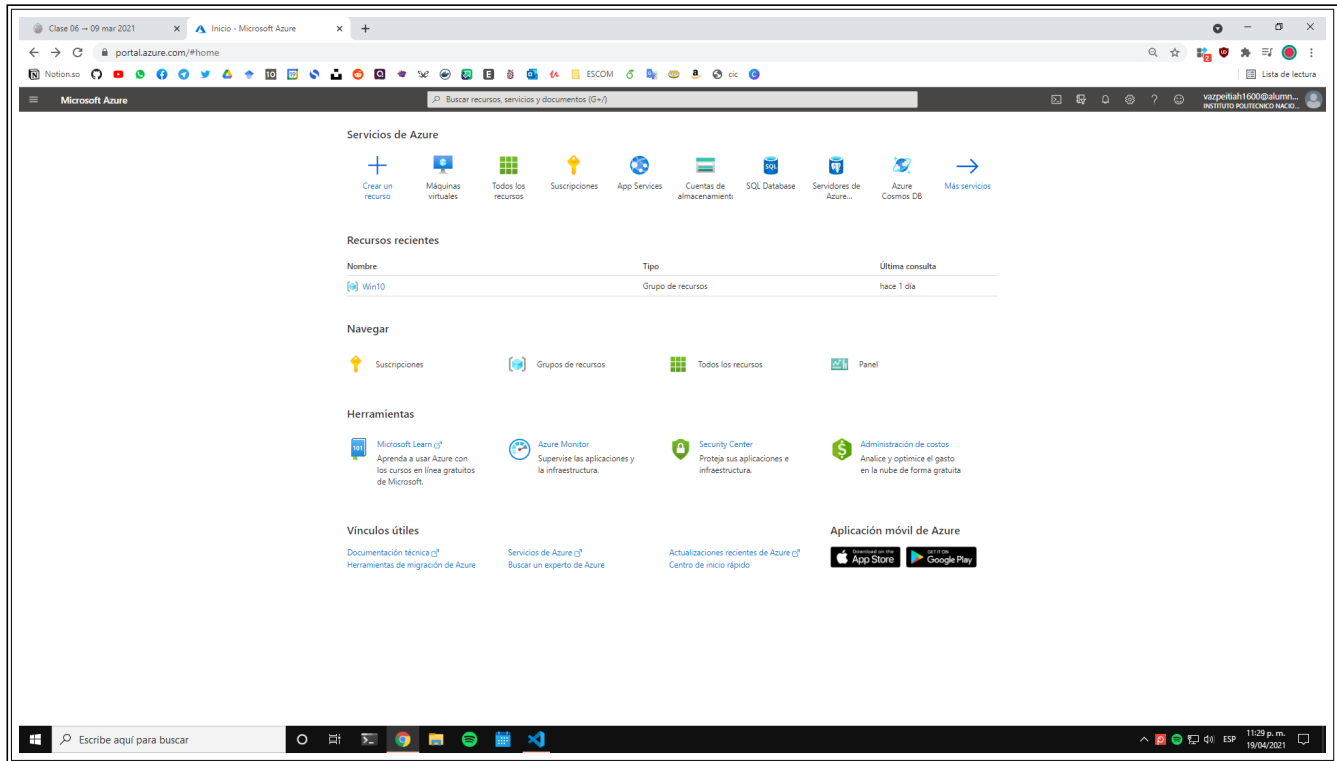


Figura 2: Creación del nodo 0: Portal de azure

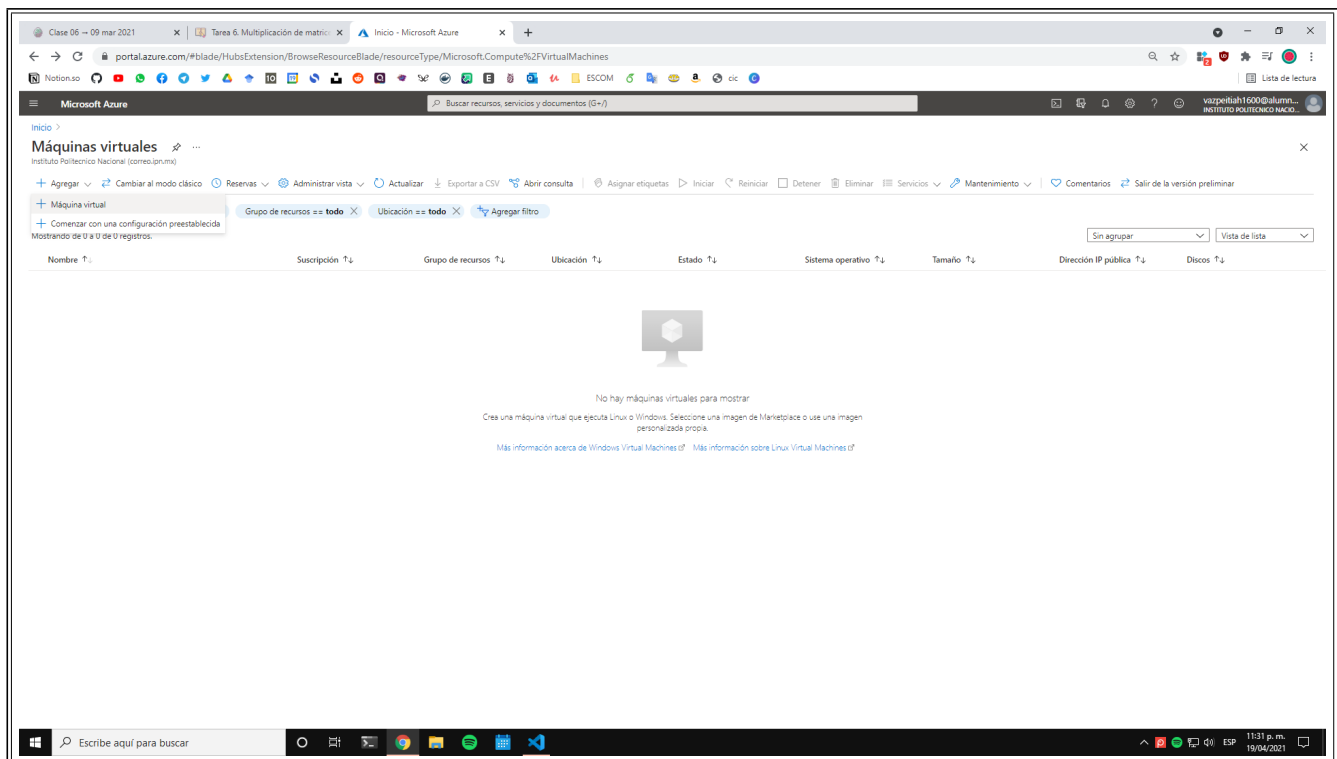


Figura 3: Creación del nodo 0: Máquinas virtuales

Microsoft Azure

Crear una máquina virtual

para organizar y administrar todos los recursos.

Suscripción * Azure para estudiantes

Grupo de recursos * UbuntuServer_group

Detalles de instancia

Nombre de máquina virtual * 2017350201-0

Región * (US) Centro-Sur de EE. UU.

Opciones de disponibilidad * No se requiere redundancia de la infraestructura

Imagen * Ubuntu Server 18.04 LTS - Gen1

Instancia de Azure de acceso puntual * ☐

Tamaño * Standard_B1s - 1 vCPU, 1 GB de memoria (176,11 MB/mem)

Cuenta de administrador

Tipo de autenticación * ☐ Clave pública SSH ☒ Contraseña

Nombre de usuario * vazeptiah

Contraseña *

Confirmar contraseña *

Reglas de puerto de entrada

Seleccione los puertos de red de máquina virtual que son accesibles desde la red internet pública. Puede especificar acceso de red más limitado o granular en la pestaña Red.

Puertos de entrada públicos * ☐ Ninguno ☒ Permitir los puertos seleccionados

Seleccionar puertos de entrada * SSH (22)

Nota: Esto permitirá que todas las direcciones IP accedan a la máquina virtual. Esto solo se recomienda para las pruebas. Use los controles avanzados de la pestaña Redes a fin de crear reglas para limitar el tráfico entrante a las direcciones IP conocidas.

Revisar y crear

Figura 4: Creación del nodo 0: Datos básicos de la máquina virtual

Microsoft Azure

Crear una máquina virtual

Datos básicos Discos Redes Administración Opciones avanzadas Etiquetas Revisar y crear

Las máquinas virtuales de Azure tienen un disco de sistema operativo y un disco temporal para el almacenamiento a corto plazo. Puede asociar discos de datos adicionales. El tamaño de la máquina virtual determina el tipo de almacenamiento que puede usar y la cantidad de datos que permiten los discos. [Más información](#)

Opciones de disco

Tipo de disco del sistema operativo * HDD estándar

Tipo de cifrado * (Predeterminado) Cifrado en reposo con una clave administrada por la pl...

Habilitar compatibilidad con Ultra Disks ☐

Discos de datos

Puede agregar y configurar discos de datos adicionales para su máquina virtual o asociar discos existentes. Esta máquina virtual también incluye un disco temporal.

LUN	Nombre	Tamaño (G...)	Tipo de disco	Almacenamiento e...
Crear y adjuntar un nuevo disco Asociar un disco existente				

Opciones avanzadas

Revisar y crear

Figura 5: Creación del nodo 0: Discos de la máquina virtual

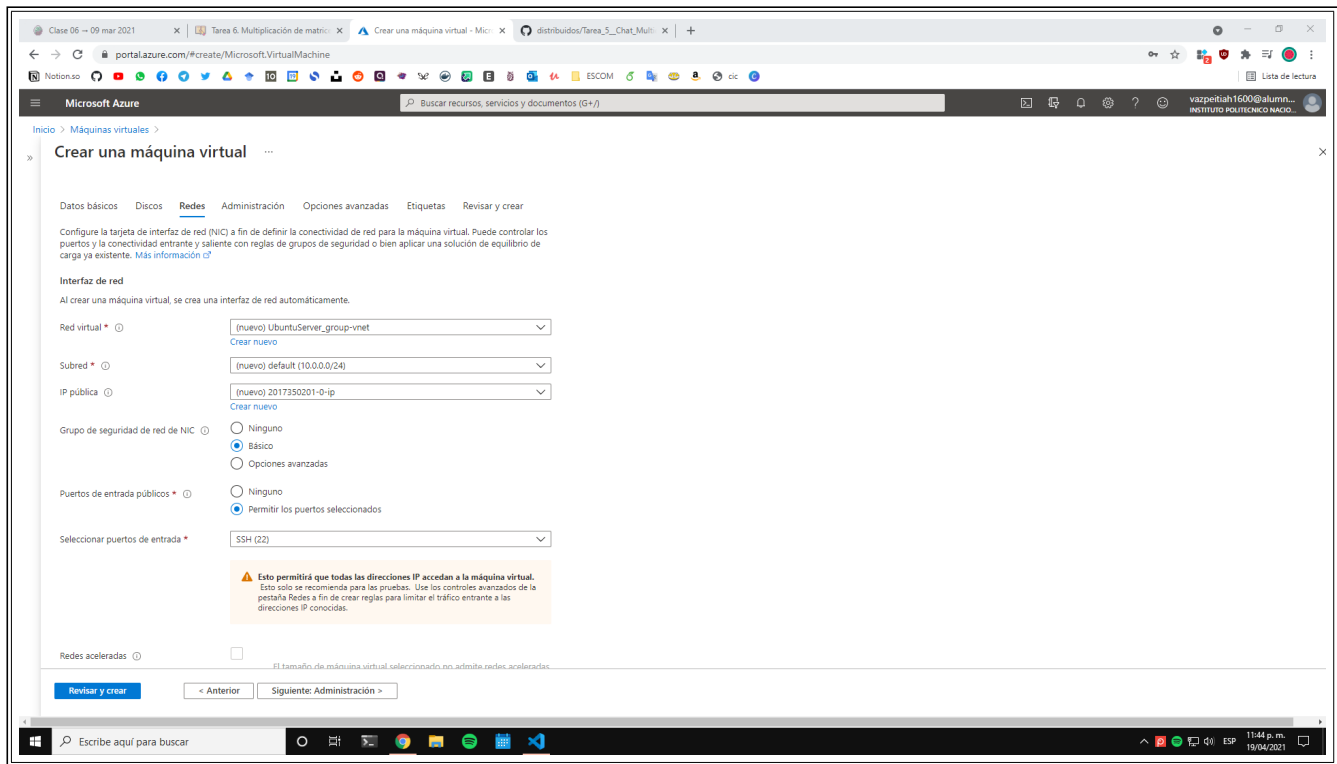


Figura 6: Creación del nodo 0: Redes de la máquina virtual

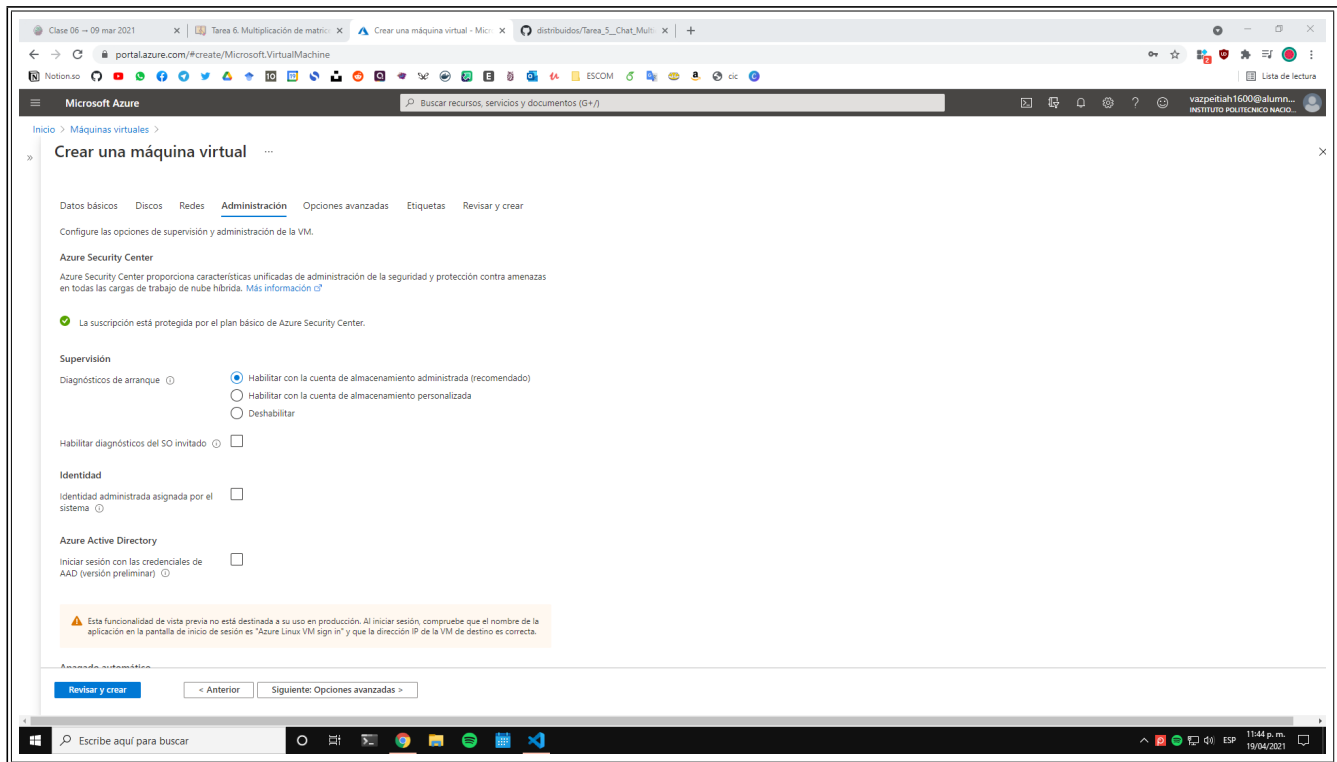


Figura 7: Creación del nodo 0: Administración de la máquina virtual

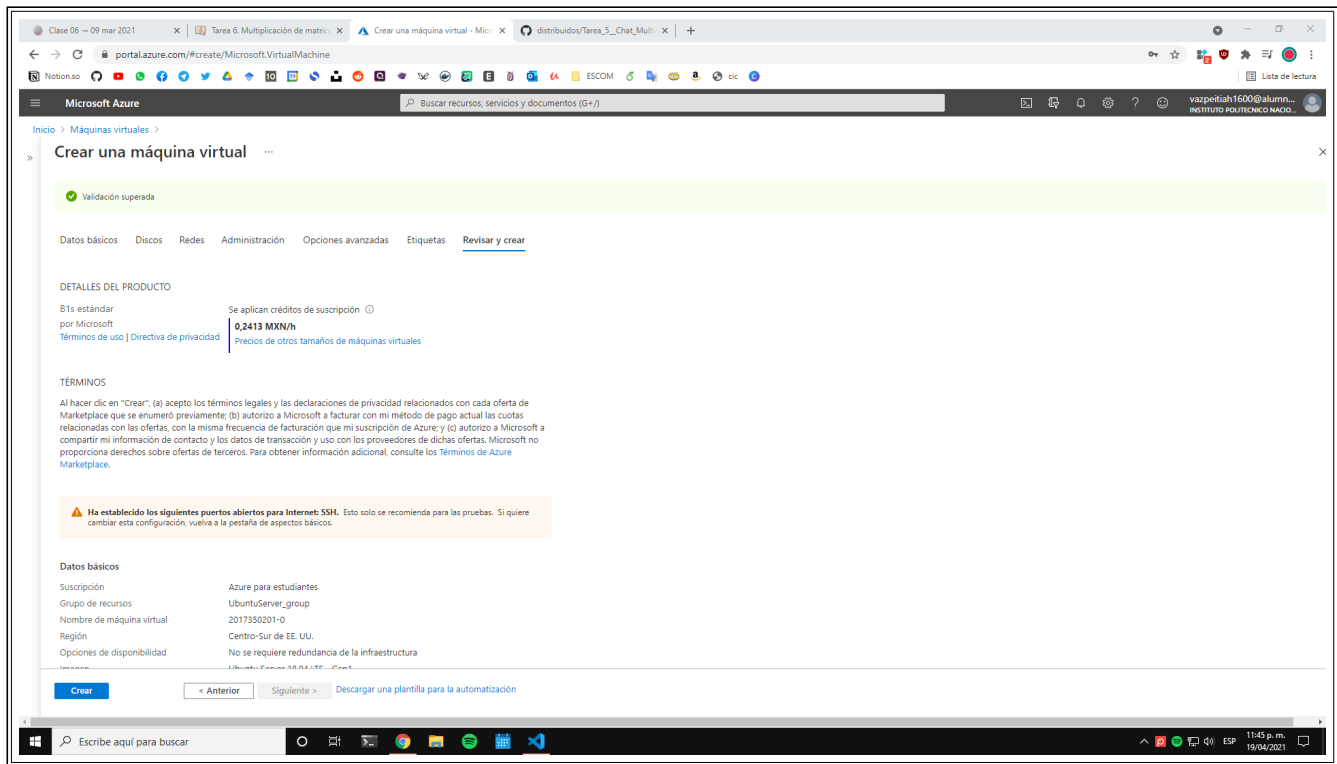


Figura 8: Creación del nodo 0: Validación superada

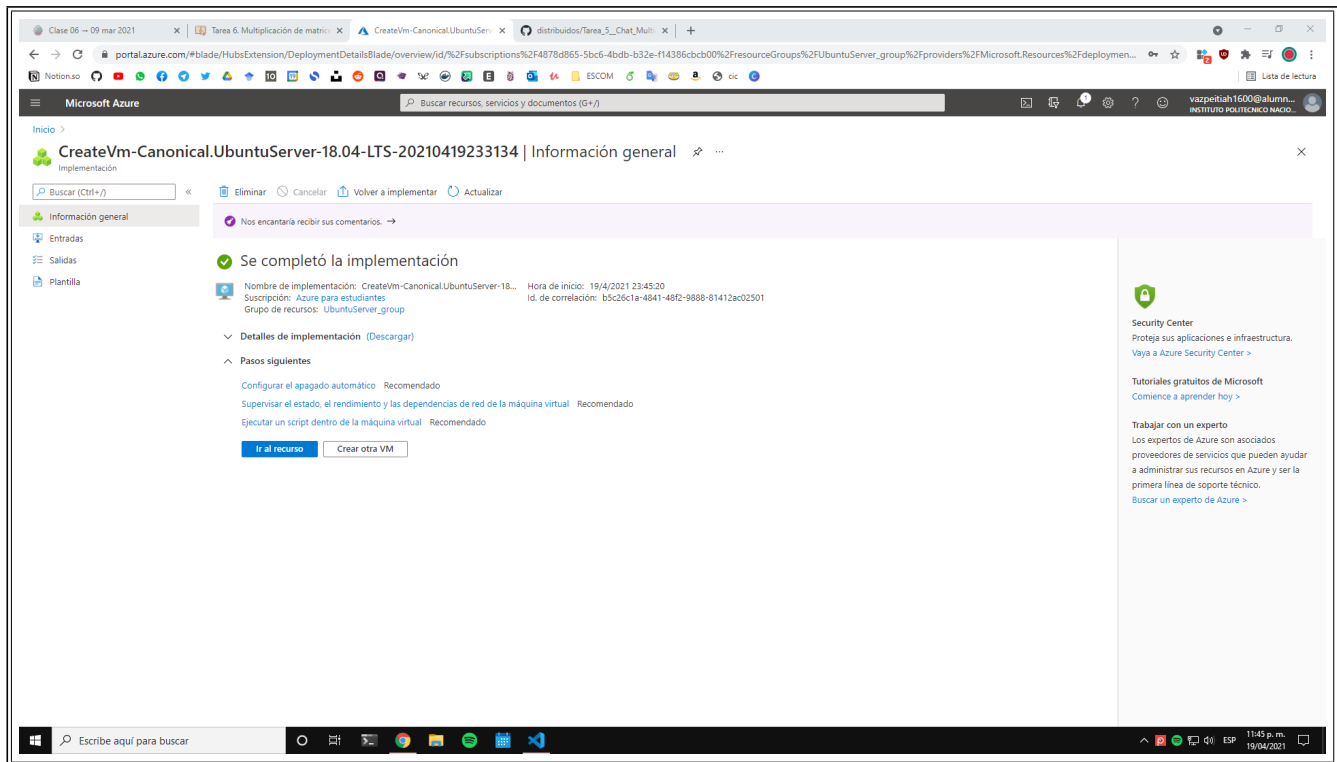


Figura 9: Creación del nodo 0: Implementación completada

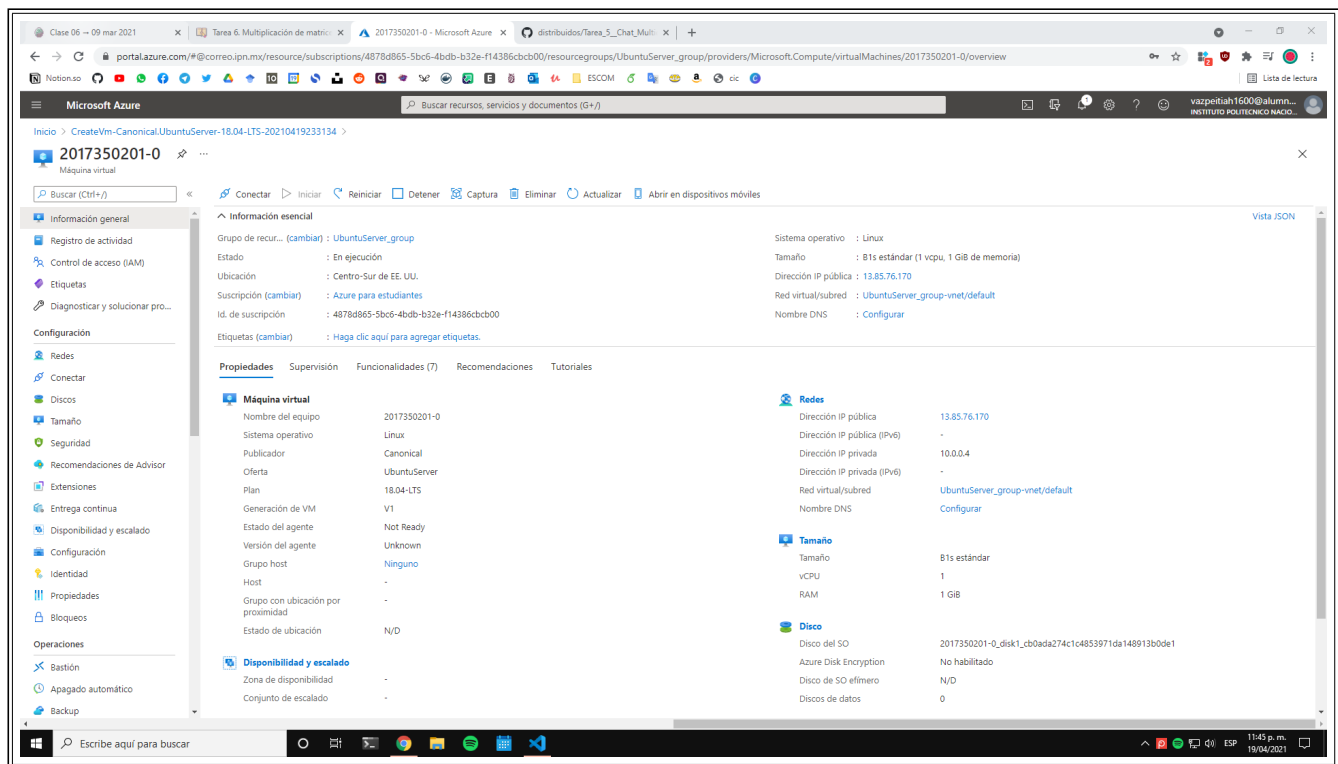


Figura 10: Creación del nodo 0: Panel de control de la máquina virtual

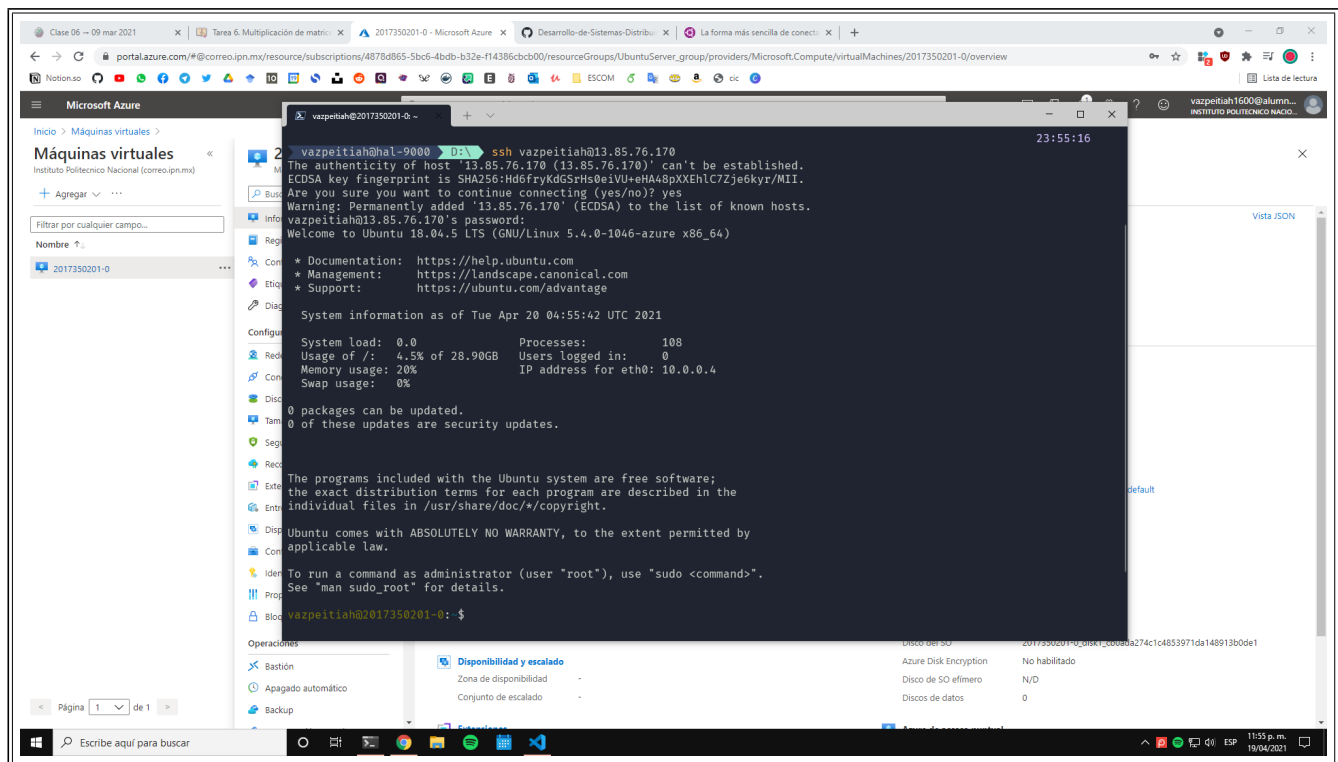


Figura 11: Creación del nodo 0: Conectarse a la máquina virtual a través de ssh

```
vazpeitiah@2017350201-0: ~$ sudo apt install default-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-headers-4.15.0-142
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  at-spi2-core ca-certificates-java default-jdk-headless default-jre default-jre-headless fontconfig-config
  fonts-dejavu-core fonts-dejavu-extra java-common libasound2 libasound2-data libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libdrm-amdgpu1 libdrm-intel1 libdrm-nouveau2
  libdrm-radeon1 libfontconfig1 libfontconfig1 libgl1 libgl1-mesa-dri libglapi-mesa libglvnd0 libglx-mesa0 libglx0
  libgraphite2-3 libharfbuzz0b libice-dev libice6 libjpeg-turbo8 libjpeg8 liblcms2-2 libllvm10 libnspr4 libnss3
  libpciaccess0 libpcscite1 libpthread-stubs0-dev libensors4 libsm-dev libsm6 libx11-dev libx11-doc libx11-xcb1
  libxau-dev libxaw7 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0 libxcb-present0 libxcb-shape0 libxcb-sync1 libxcb1-dev
  libxcomposite1 libxdamage1 libxdmcp-dev libxfixes3 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2 libxrender1
  libxshmfence1 libxt-dev libxt6 libxtst6 libxv1 libxxf86dgal libxxf86vm1 openjdk-11-jdk openjdk-11-jdk-headless
  openjdk-11-jre openjdk-11-jre-headless x11-common x11-utils x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
Suggested packages:
  libasound2-plugins alsa-utils libice-doc liblcms2-utils pcsd lm-sensors libsm-doc libxcb-doc libxt-doc
  openjdk-11-demo openjdk-11-source visualvm libnss-mdns fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
  | fonts-wqy-zenhei fonts-indic mesa-utils
The following NEW packages will be installed:
  at-spi2-core ca-certificates-java default-jdk default-jdk-headless default-jre default-jre-headless fontconfig-config
  fonts-dejavu-core fonts-dejavu-extra java-common libasound2 libasound2-data libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libdrm-amdgpu1 libdrm-intel1 libdrm-nouveau2
  libdrm-radeon1 libfontconfig1 libfontconfig1 libgl1 libgl1-mesa-dri libglapi-mesa libglvnd0 libglx-mesa0 libglx0
  libgraphite2-3 libharfbuzz0b libice-dev libice6 libjpeg-turbo8 libjpeg8 liblcms2-2 libllvm10 libnspr4 libnss3
  libpciaccess0 libpcscite1 libpthread-stubs0-dev libensors4 libsm-dev libsm6 libx11-dev libx11-doc libx11-xcb1
  libxau-dev libxaw7 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0 libxcb-present0 libxcb-shape0 libxcb-sync1 libxcb1-dev
  libxcomposite1 libxdamage1 libxdmcp-dev libxfixes3 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2 libxrender1
  libxshmfence1 libxt-dev libxt6 libxtst6 libxv1 libxxf86dgal libxxf86vm1 openjdk-11-jdk openjdk-11-jdk-headless
  openjdk-11-jre openjdk-11-jre-headless x11-common x11-utils x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
0 upgraded, 87 newly installed, 0 to remove and 0 not upgraded.
Need to get 273 MB of archives.
After this operation, 748 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Figura 12: Creación del nodo 0: Instalar Java en la máquina virtual

```
vazpeitiah@2017350201-0: ~$ git clone https://github.com/vazpeitiah/distribuidos
Cloning into 'distribuidos'...
remote: Enumerating objects: 223, done.
remote: Counting objects: 100% (223/223), done.
remote: Compressing objects: 100% (208/208), done.
remote: Total 223 (delta 78), reused 72 (delta 11), pack-reused 0
Receiving objects: 100% (223/223), 15.11 MiB | 28.28 MiB/s, done.
Resolving deltas: 100% (78/78), done.
vazpeitiah@2017350201-0: ~$ cd distribuidos/tarea6/
vazpeitiah@2017350201-0: ~$ ls -al
total 28
drwxrwxr-x 2 vazpeitiah vazpeitiah 4096 Apr 20 05:04 .
drwxrwxr-x 10 vazpeitiah vazpeitiah 4096 Apr 20 05:04 ..
-rw-rw-r-- 1 vazpeitiah vazpeitiah 661 Apr 20 05:04 ClaseRMI.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 2767 Apr 20 05:04 ClienteRMI.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 190 Apr 20 05:04 InterfacarMI.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 2847 Apr 20 05:04 MultMatrix.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 276 Apr 20 05:04 ServidorRMI.java
vazpeitiah@2017350201-0: ~$
```

Figura 13: Creación del nodo 0: Cargar el código fuente de la tarea, en la máquina virtual

2.2. Instalación y configuración de la máquina virtual del nodo 1

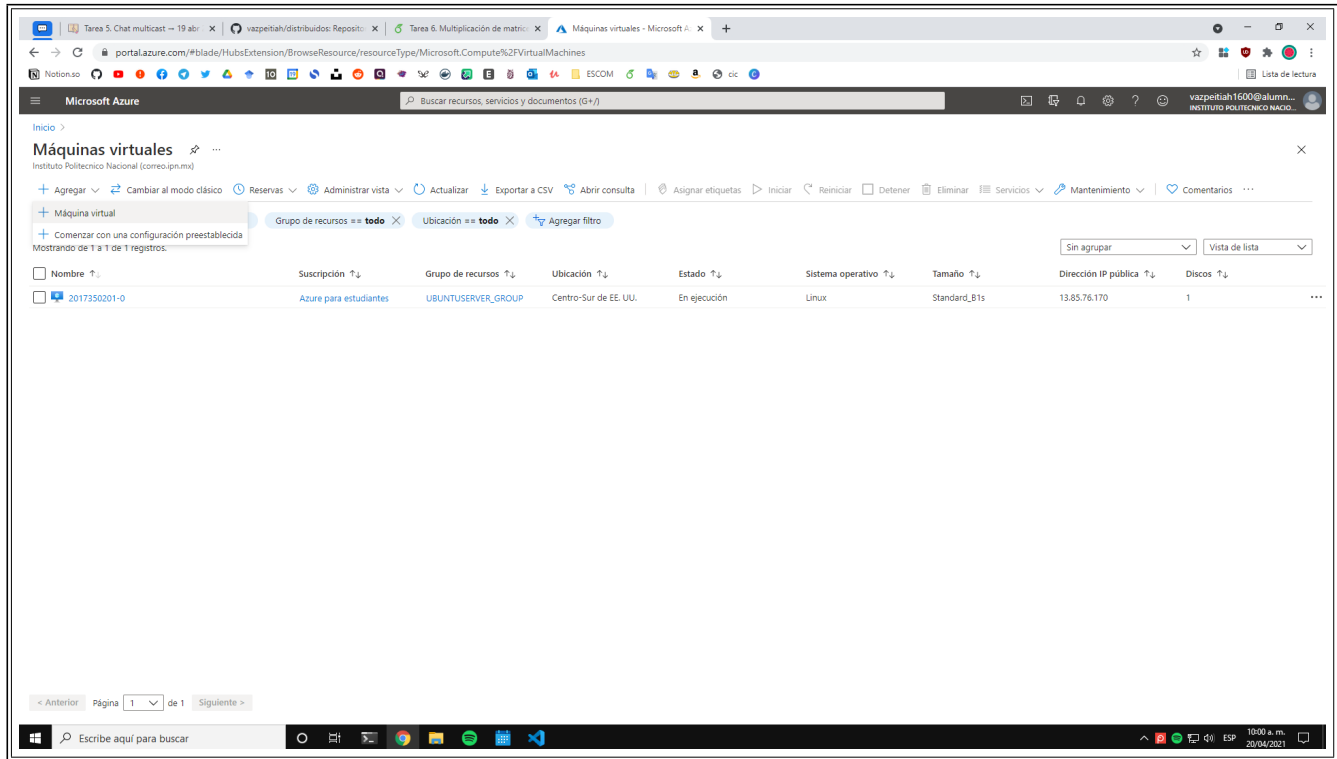


Figura 14: Creación del nodo 1: Máquinas virtuales

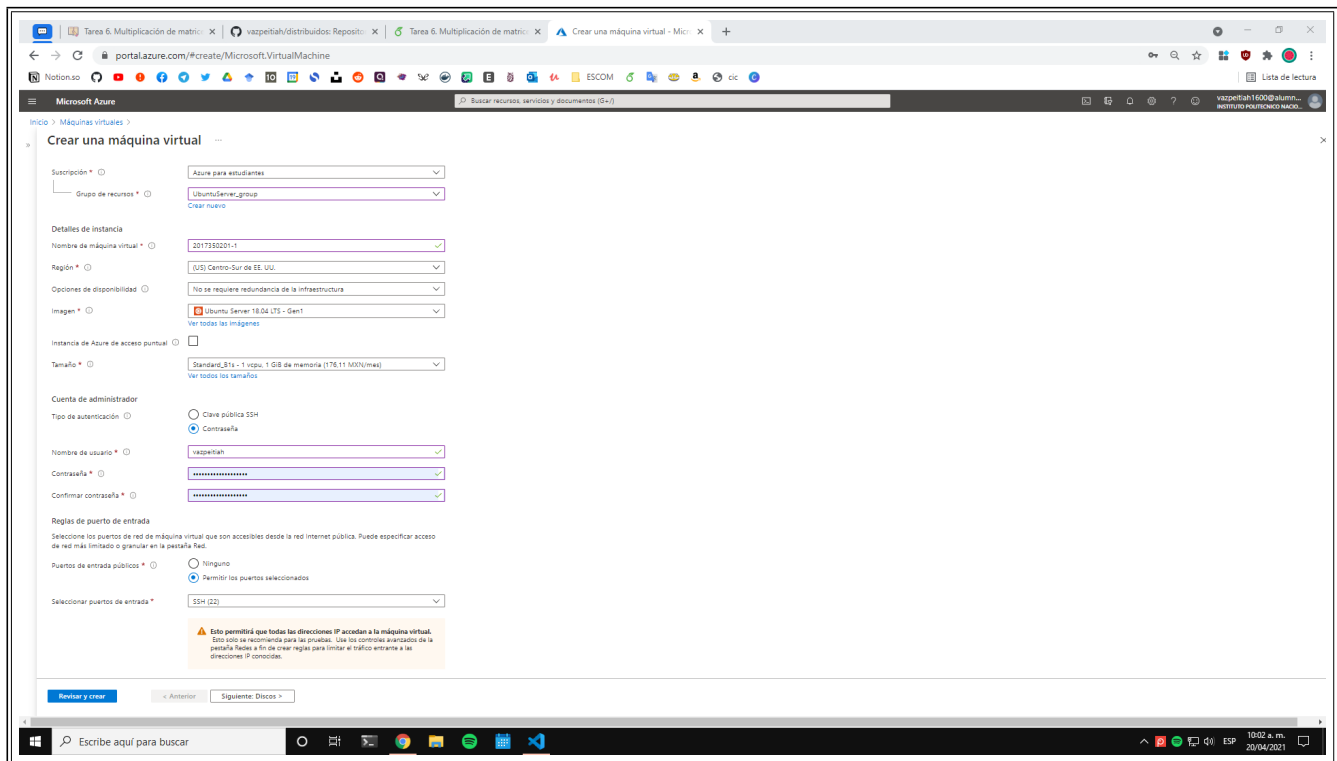


Figura 15: Creación del nodo 1: Datos básicos de la máquina virtual

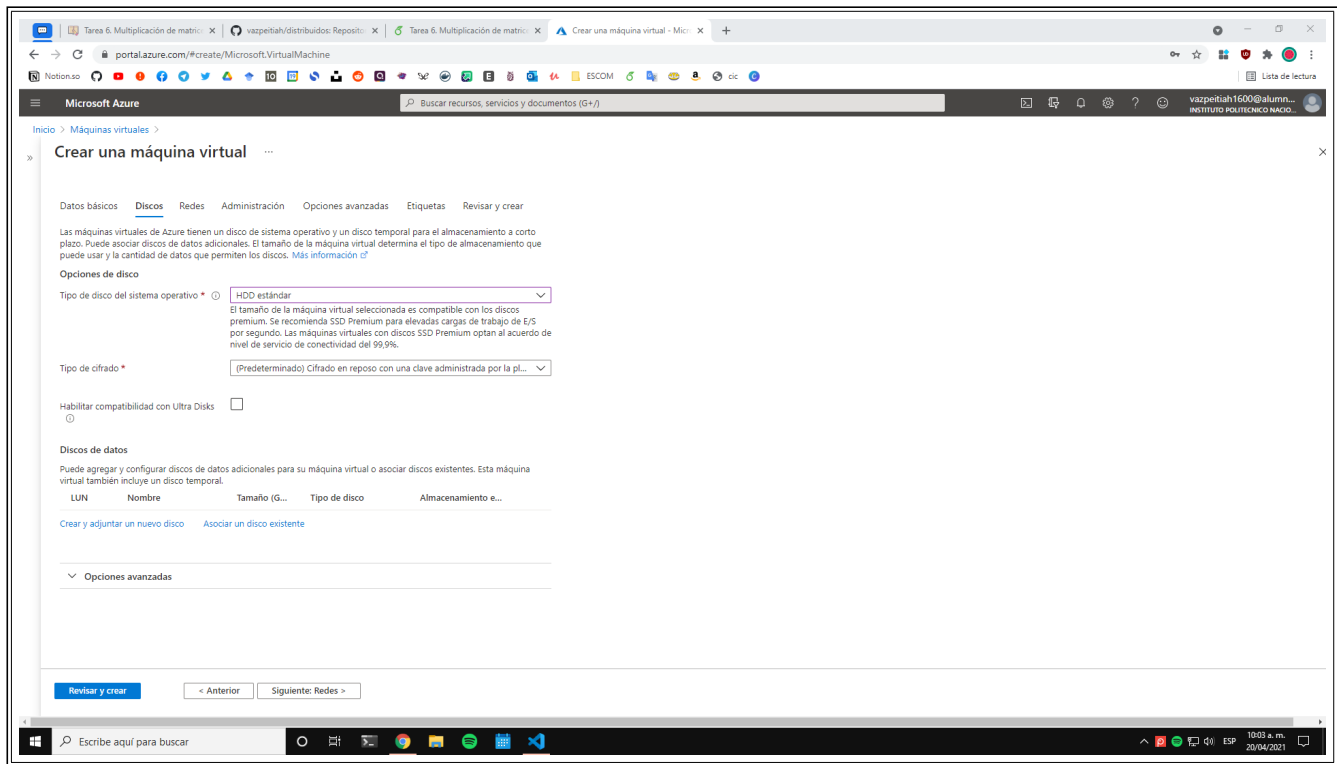


Figura 16: Creación del nodo 1: Discos de la máquina virtual

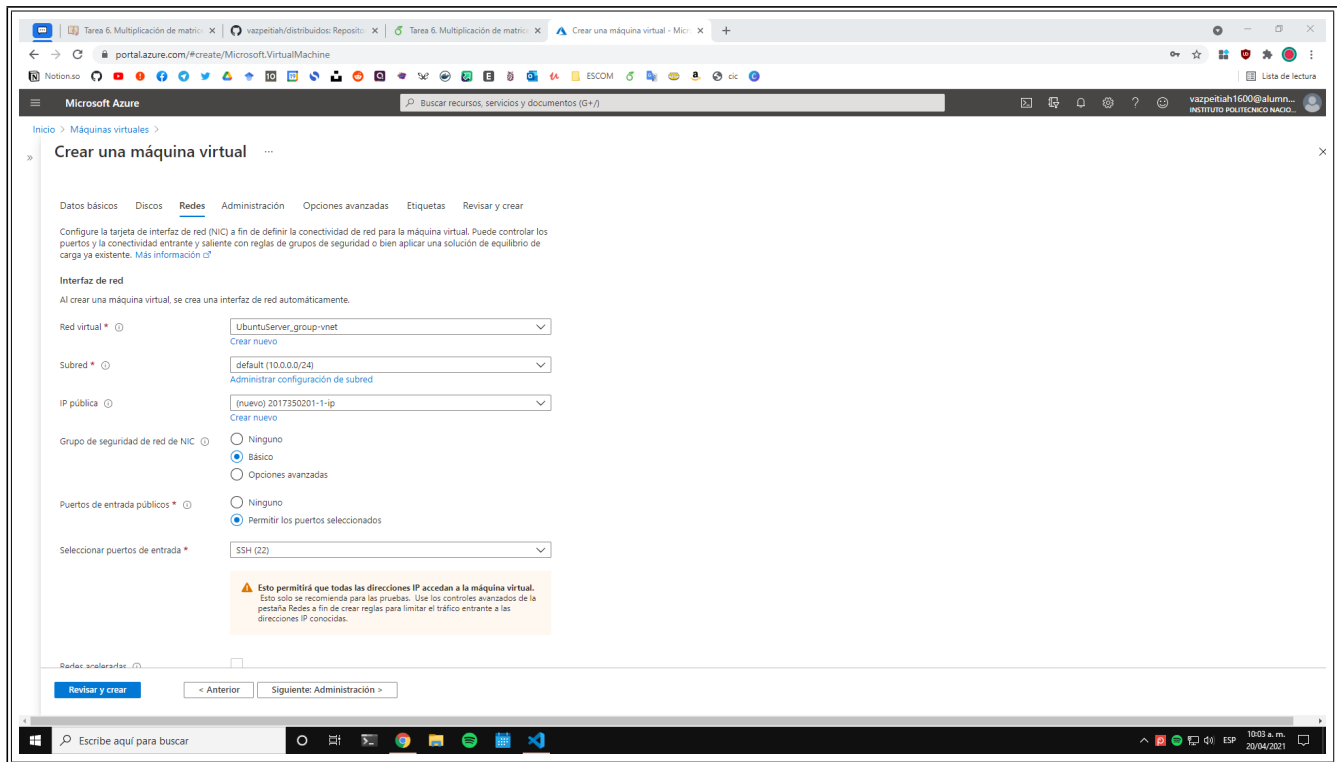


Figura 17: Creación del nodo 1: Redes de la máquina virtual

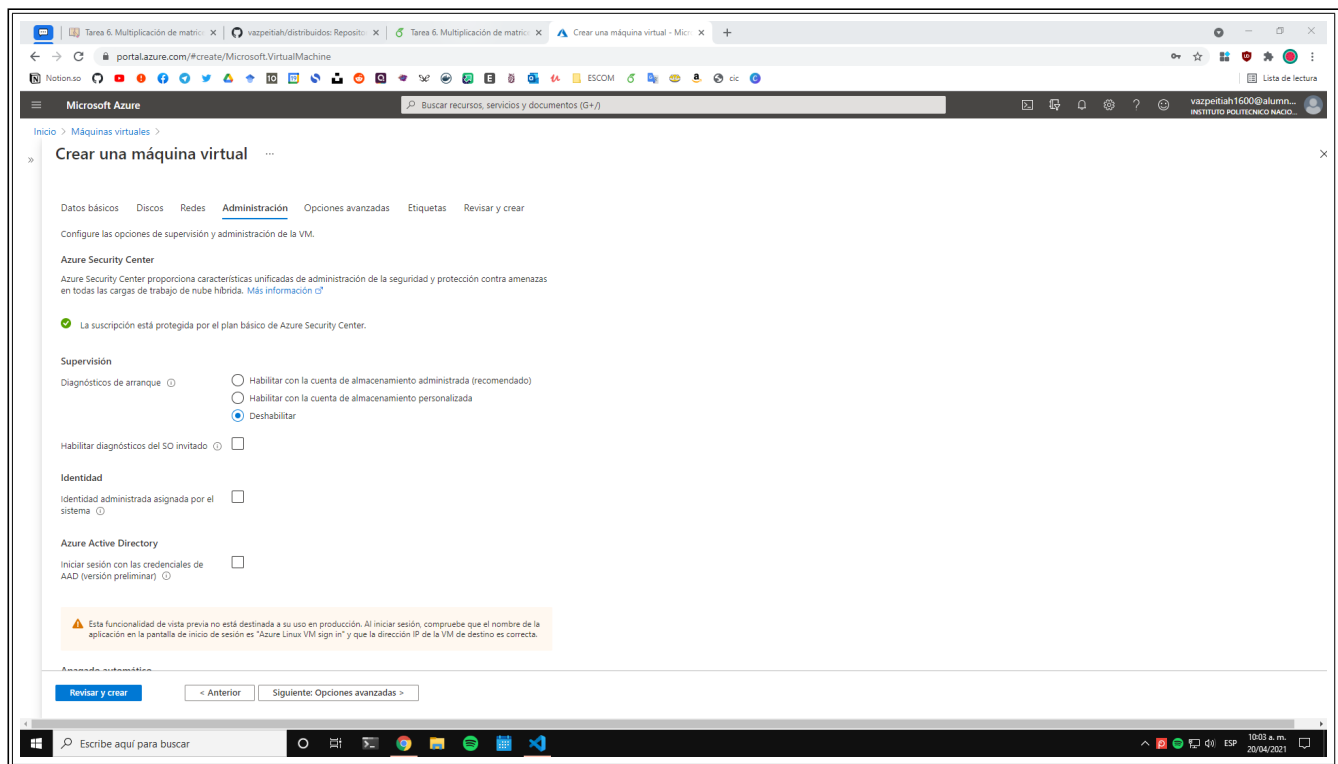


Figura 18: Creación del nodo 1: Administración de la máquina virtual

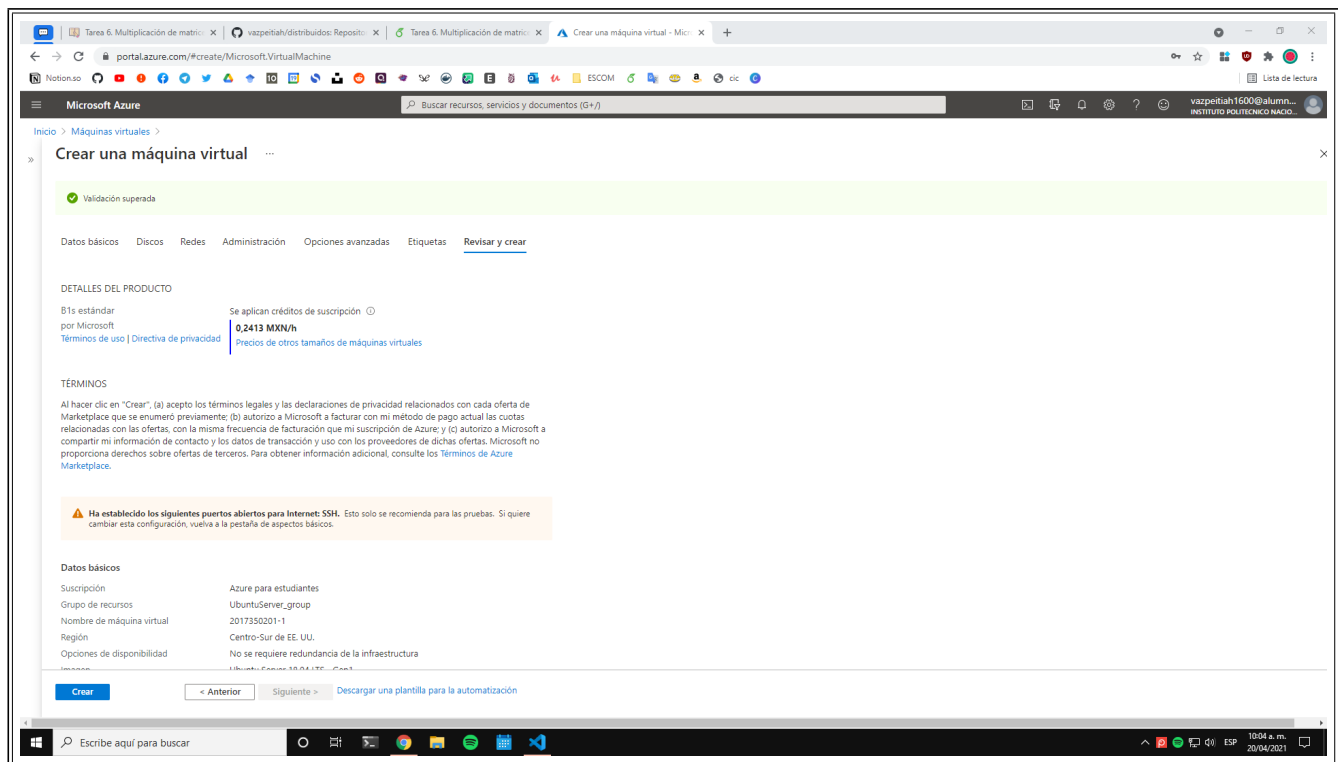


Figura 19: Creación del nodo 1: Validación superada

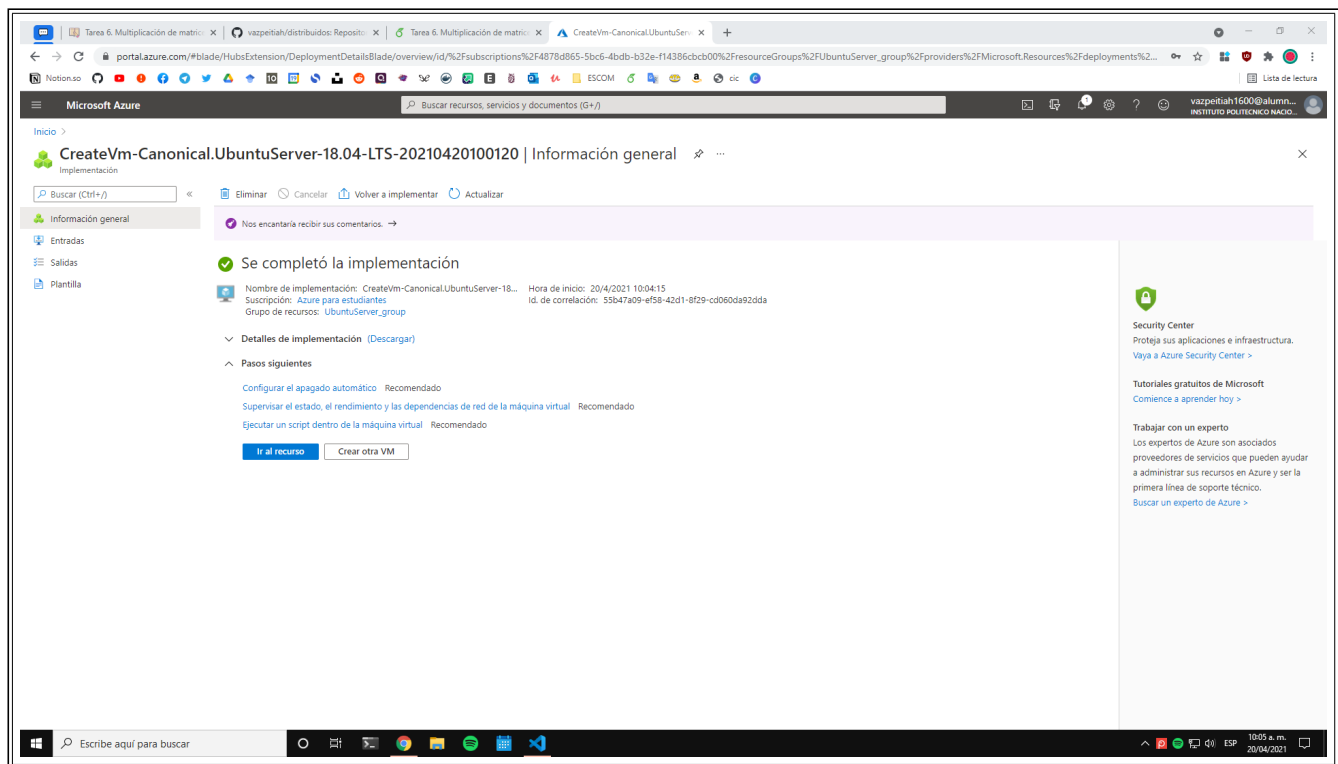


Figura 20: Creación del nodo 1: Implementación completada

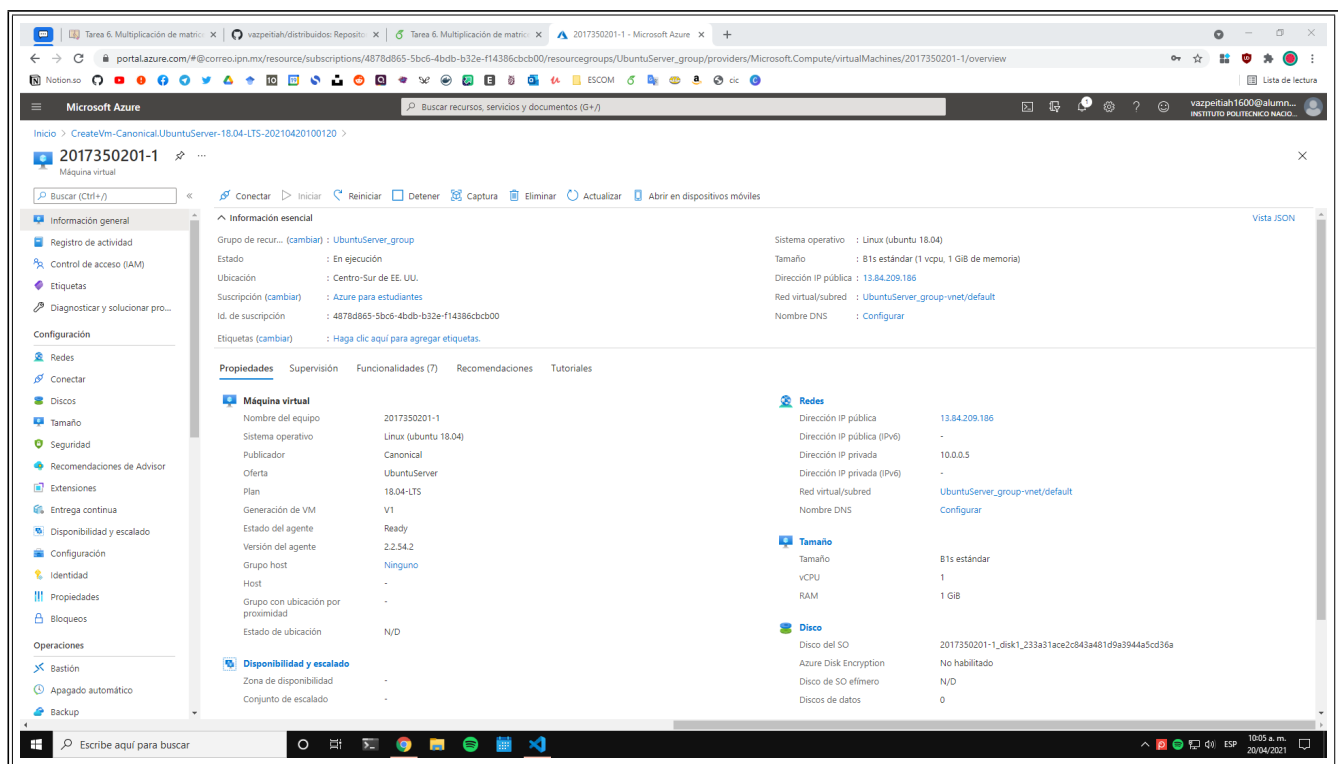


Figura 21: Creación del nodo 1: Panel de control de la máquina virtual

```
vazpeitiah@2017350201-1: ~$ ssh vazpeitiah@13.84.209.186
vazpeitiah@13.84.209.186's password:
Permission denied, please try again.
vazpeitiah@13.84.209.186's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1046-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Apr 20 15:07:39 UTC 2021

System load:  0.12          Processes:    112
Usage of /:   4.5% of 28.9GB Users logged in:   0
Memory usage: 20%          IP address for eth0: 10.0.0.5
Swap usage:   0%

0 packages can be updated.
0 of these updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

vazpeitiah@2017350201-1:~$
```

Figura 22: Creación del nodo 1: Conectarse a la máquina virtual a través de ssh

```
vazpeitiah@2017350201-1:~$ sudo apt install default-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-headers-4.15.0-142
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  at-spi2-core ca-certificates-java default-jdk-headless default-jre default-jre-headless fontconfig-config fonts-dejavu-core fonts-dejavu-extra java-common libasound2 libasound2-data
  libatk-bridge2.0-0 libatk-wrapper-java libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libdrm-amdgpu libdrm-intel1 libdrm-nouveau2 libdrm-radeon1 libfontconfig1
  libfontenc1 libgif7 libgl1 libgl1-mesa-dri libglapi-mesa libglvnd0 libglx-mesa0 libglx0 libgraphite2-3 libharfbuzz0b libice-dev libice6 libjpeg-turbo8 libjpeg8 liblcms2-2 libllvml10
  libnspr4 libnss3 libpciaccess0 libpcsc-lite libpthread-stubs0-dev libsensors4 libsm-dev libsm6 libx11-dev libx11-doc libx11-xcb libxau-dev libxaw7 libxcb-dri2-0 libxcb-dri3-0
  libxcb-glx0 libxcb-present0 libxcb-shape0 libxcb-sync1 libxcb1-dev libxcomposite1 libxdamage1 libxdmcp-dev libxfixes3 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2 libxrender1
  libxshmfence1 libxt-dev libxt6 libxtst6 libxv1 libxxf86dga1 libxxf86vm1 openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11-common x11-utils
  x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  libasound2-plugins alsa-utils libice-doc liblcms2-utils pcsd lm-sensors libsm-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source visualvm libnss-mdns fonts-ipafont-gothic
  fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic mesa-utils
The following NEW packages will be installed:
  at-spi2-core ca-certificates-java default-jdk default-jdk-headless default-jre default-jre-headless fontconfig-config fonts-dejavu-core fonts-dejavu-extra java-common libasound2
  libasound2-data libatk-bridge2.0-0 libatk-wrapper-java libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libdrm-amdgpu libdrm-intel1 libdrm-nouveau2 libdrm-radeon1
  libfontconfig1 libfontenc1 libgif7 libgl1 libgl1-mesa-dri libglapi-mesa libglvnd0 libglx-mesa0 libglx0 libgraphite2-3 libharfbuzz0b libice-dev libice6 libjpeg-turbo8 libjpeg8
  liblcms2-2 libllvml10 libnspr4 libnss3 libpciaccess0 libpcsc-lite libpthread-stubs0-dev libsensors4 libsm-dev libsm6 libx11-dev libx11-doc libx11-xcb libxau-dev libxaw7 libxcb-dri2-0
  libxcb-dri3-0 libxcb-glx0 libxcb-present0 libxcb-shape0 libxcb-sync1 libxcb1-dev libxcomposite1 libxdamage1 libxdmcp-dev libxfixes3 libxft2 libxi6 libxinerama1 libxmu6 libxpm4
  libxrandr2 libxrender1 libxshmfence1 libxt-dev libxt6 libxtst6 libxv1 libxxf86dga1 libxxf86vm1 openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11-common
  x11-utils x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 87 newly installed, 0 to remove and 0 not upgraded.
Need to get 273 MB of archives.
After this operation, 748 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Figura 23: Creación del nodo 1: Instalar Java en la máquina virtual

```
vazpeitiah@2017350201-1: ~$ git clone https://github.com/vazpeitiah/distribuidos
Cloning into 'distribuidos'...
remote: Enumerating objects: 236, done.
remote: Counting objects: 100% (236/236), done.
remote: Compressing objects: 100% (214/214), done.
remote: Total 236 (delta 86), reused 84 (delta 18), pack-reused 0
Receiving objects: 100% (236/236), 15.11 MiB | 33.13 MiB/s, done.
Resolving deltas: 100% (86/86), done.
vazpeitiah@2017350201-1: ~$ cd distribuidos/tarea6
vazpeitiah@2017350201-1: ~/distribuidos/tarea6$ ls -l
total 20
-rw-rw-r-- 1 vazpeitiah vazpeitiah 670 Apr 20 15:12 ClaseRMI.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 2717 Apr 20 15:12 ClienteRMI.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 195 Apr 20 15:12 InterfaceRMI.java
-rw-rw-r-- 1 vazpeitiah vazpeitiah 246 Apr 20 15:12 Makefile
-rw-rw-r-- 1 vazpeitiah vazpeitiah 276 Apr 20 15:12 ServidorRMI.java
vazpeitiah@2017350201-1: ~/distribuidos/tarea6$ cat Makefile
JC = javac
.SUFFIXES: .java .class
.java.class:
$(JC) %.java

CLASSES = \
InterfaceRMI.java \
ClaseRMI.java \
ClienteRMI.java \
ServidorRMI.java

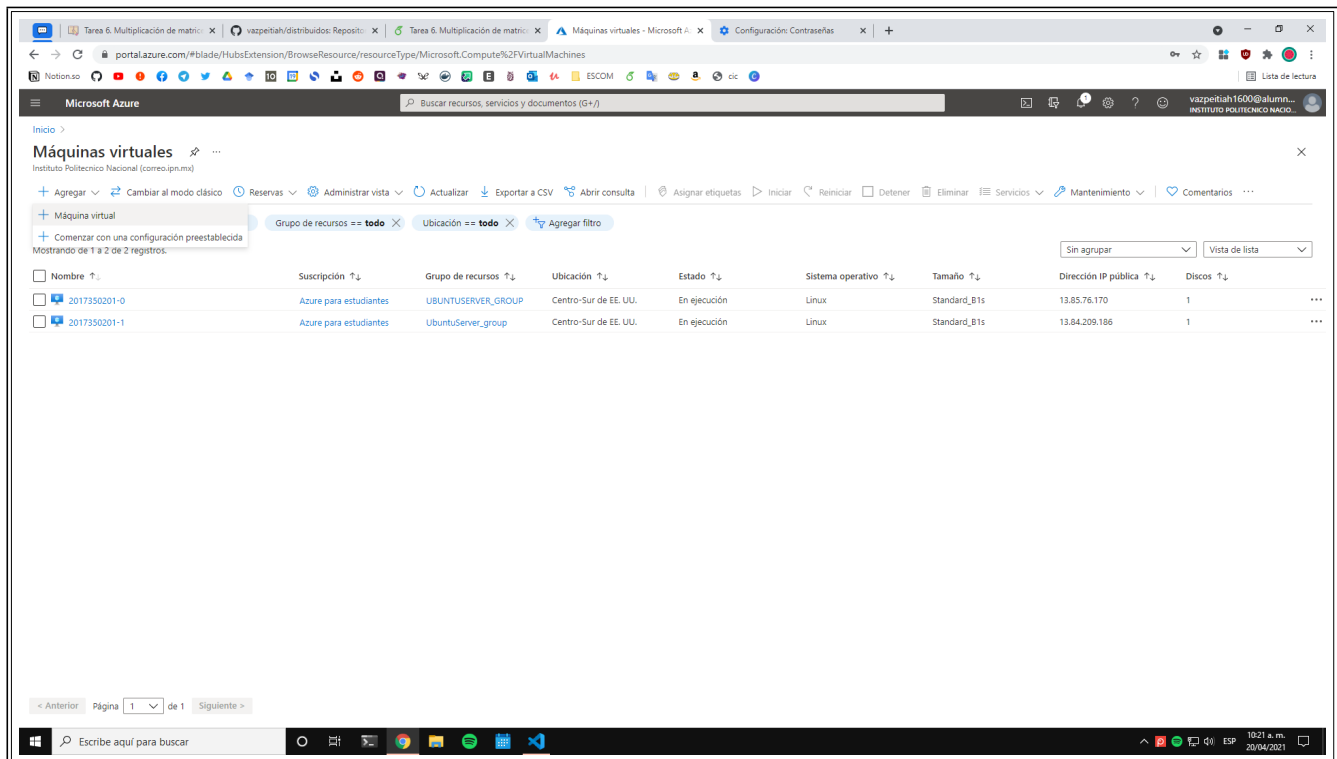
default: classes

classes: $(CLASSES:.java=.class)

clean:
rm *.class
vazpeitiah@2017350201-1: ~/distribuidos/tarea6$
```

Figura 24: Creación del nodo 1: Cargar el código fuente de la tarea, en la máquina virtual

2.3. Instalación y configuración de la máquina virtual del nodo 2



Nombre	Suscripción	Grupo de recursos	Ubicación	Estado	Sistema operativo	Tamaño	Dirección IP pública	Discos
2017350201-0	Azure para estudiantes	UBUNTUSERVER_GROUP	Centro-Sur de EE. UU.	En ejecución	Linux	Standard_B1s	13.85.76.170	1
2017350201-1	Azure para estudiantes	UbuntuServer_group	Centro-Sur de EE. UU.	En ejecución	Linux	Standard_B1s	13.84.209.186	1

Figura 25: Creación del nodo 2: Máquinas virtuales

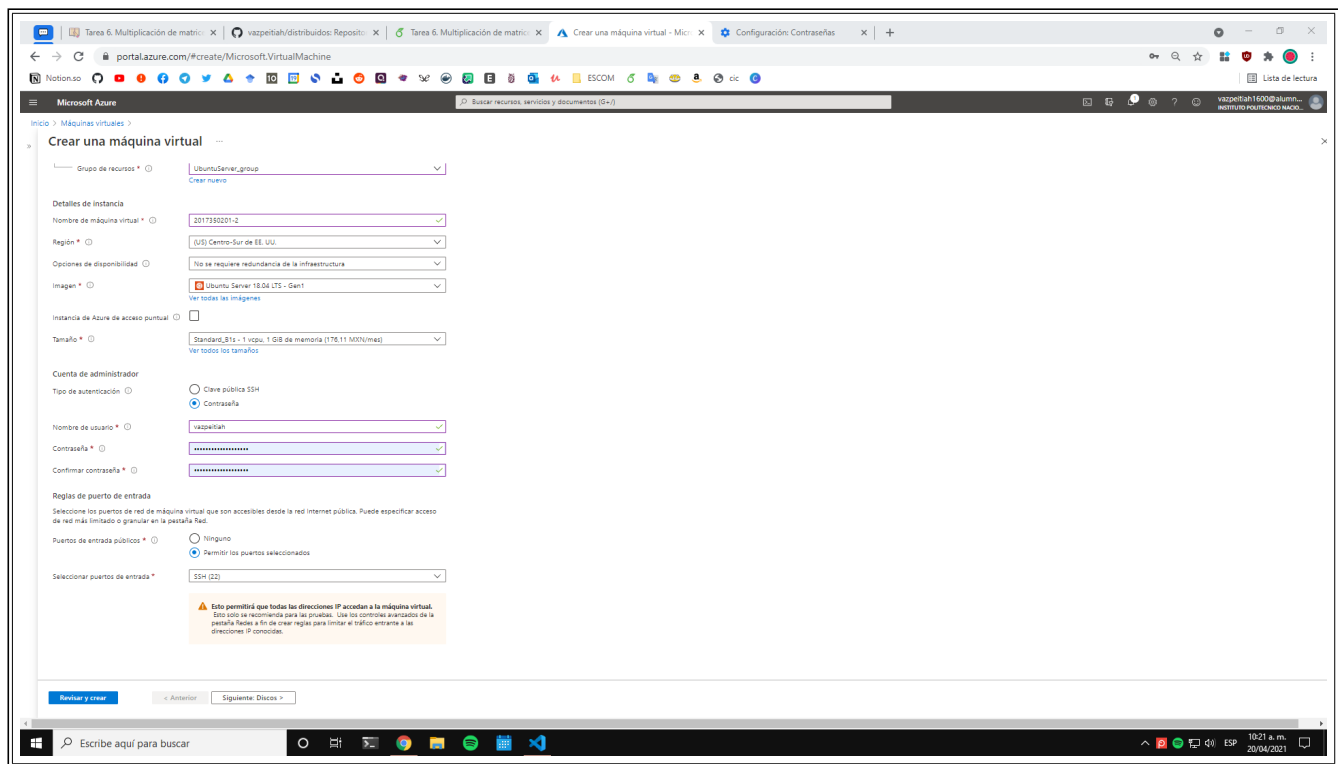


Figura 26: Creación del nodo 2: Datos básicos de la máquina virtual

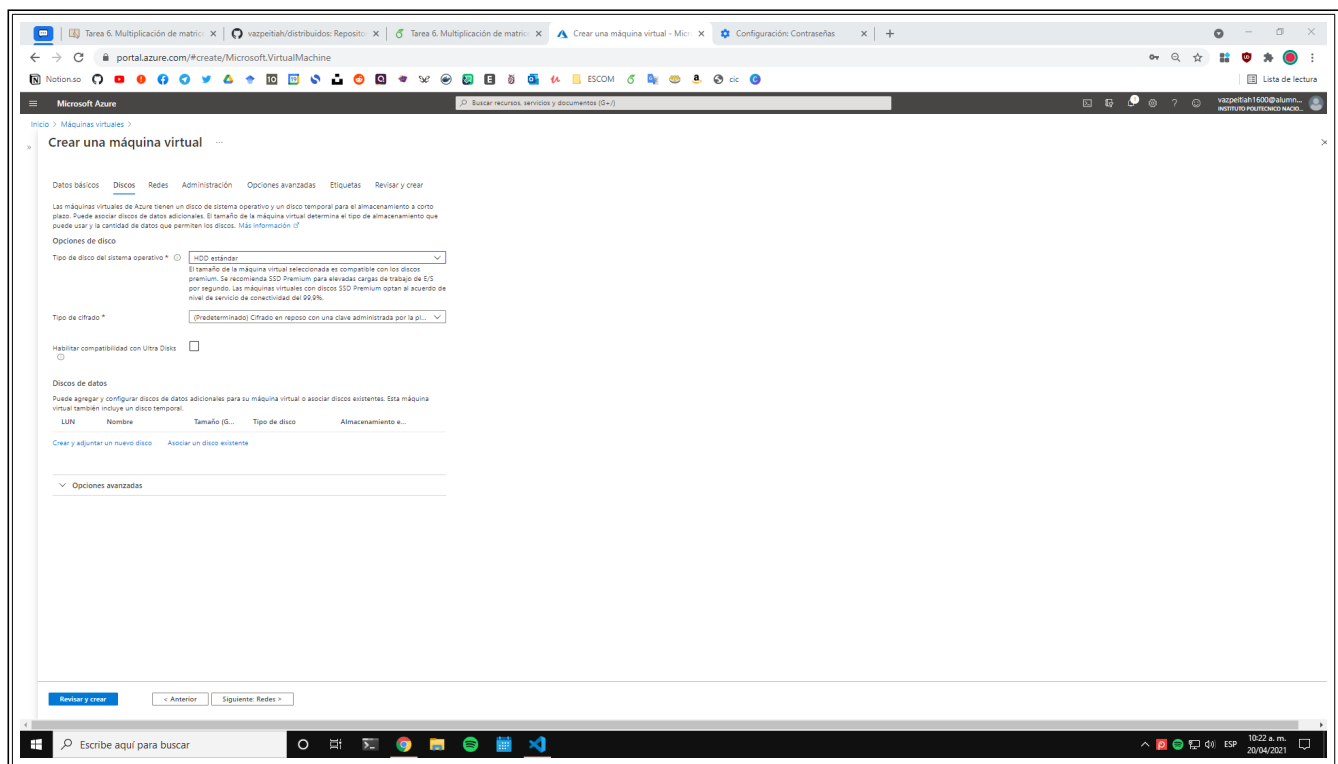


Figura 27: Creación del nodo 2: Discos de la máquina virtual

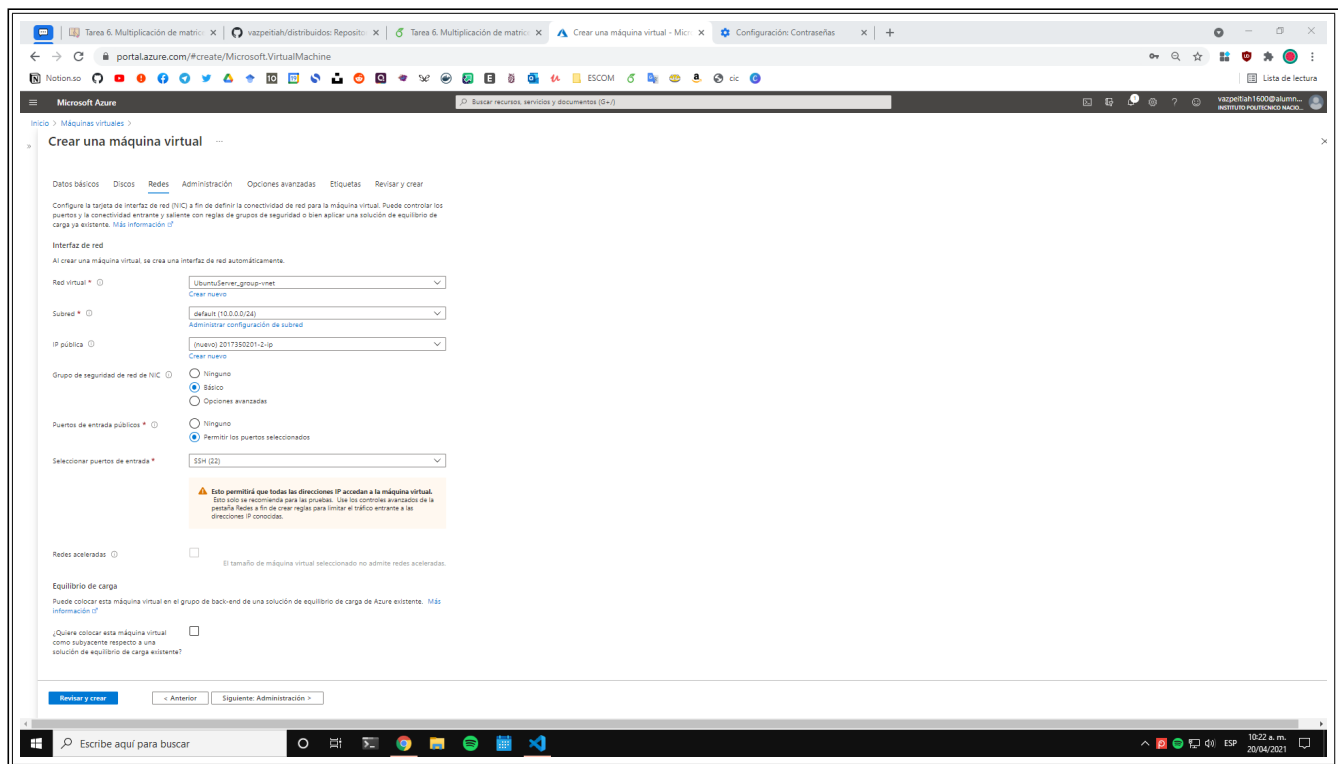


Figura 28: Creación del nodo 2: Redes de la máquina virtual

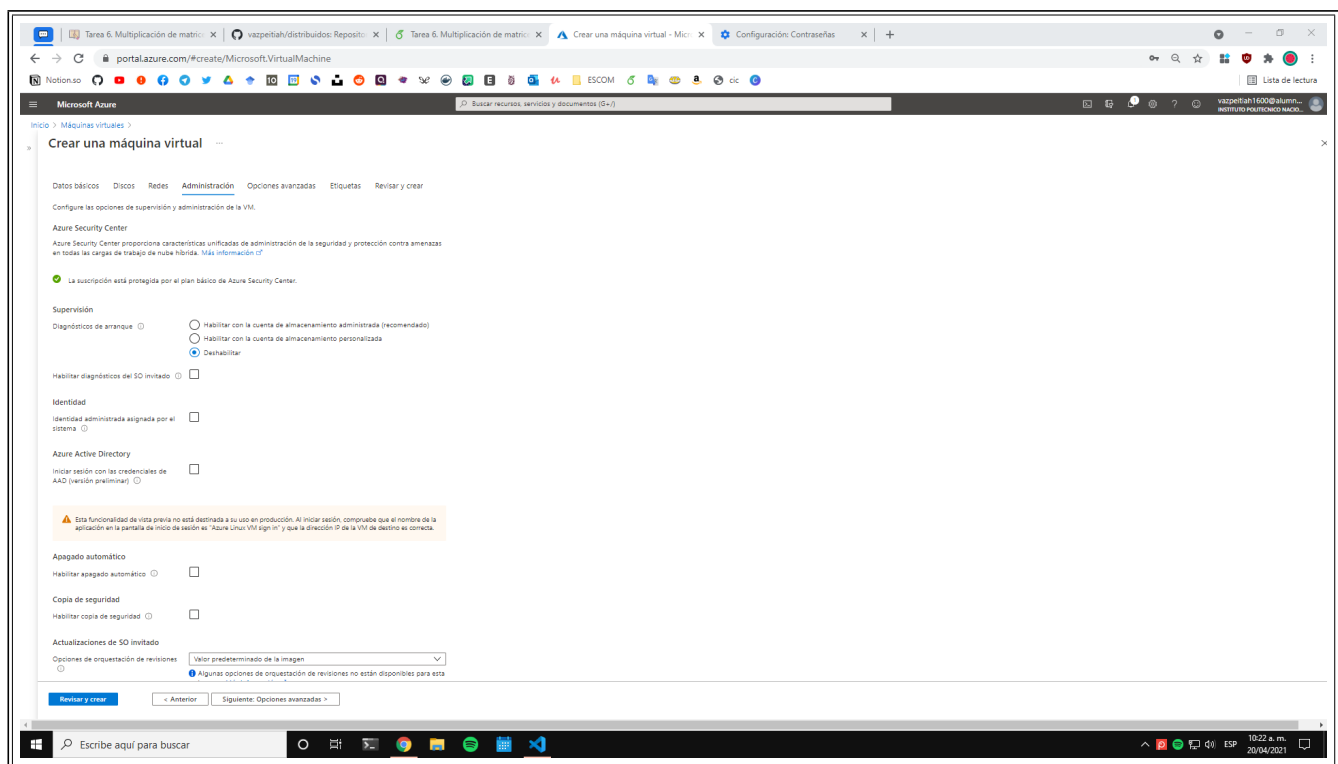


Figura 29: Creación del nodo 2: Administración de la máquina virtual

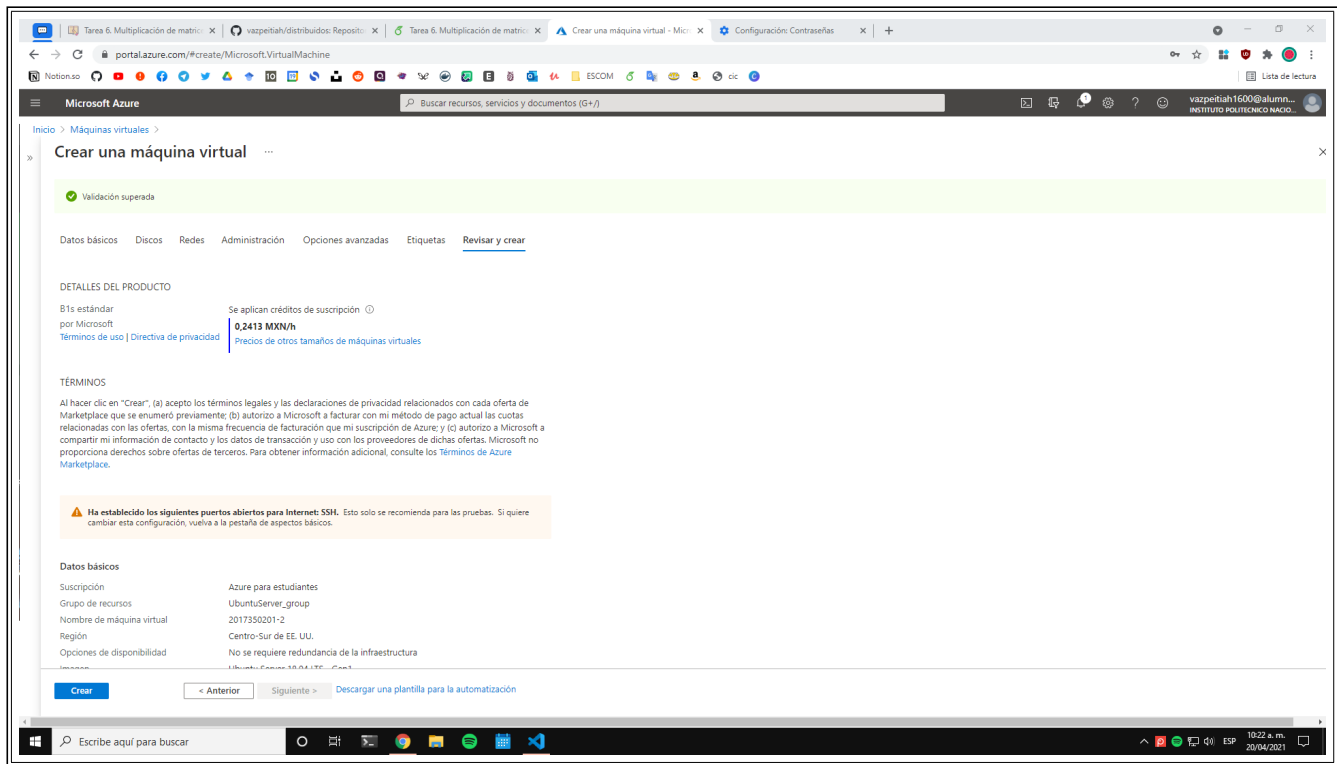


Figura 30: Creación del nodo 2: Validación superada

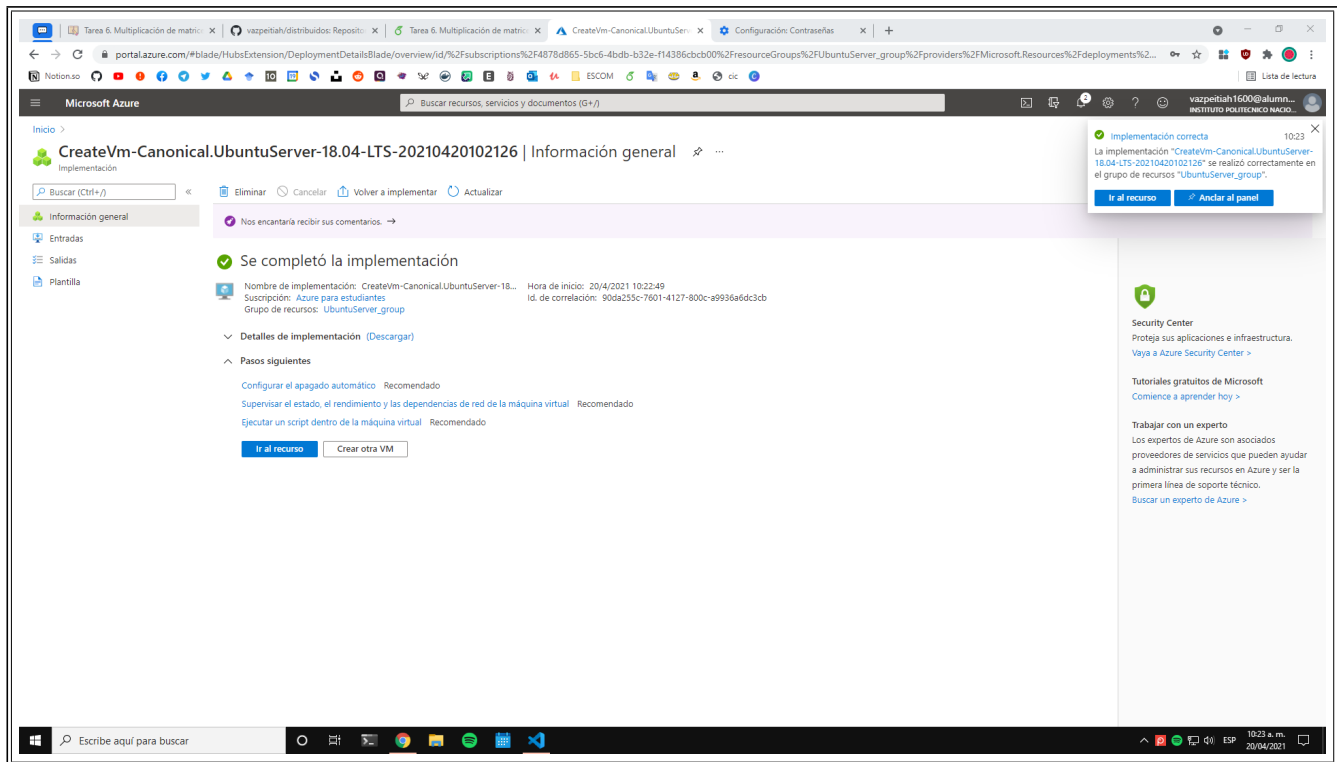


Figura 31: Creación del nodo 2: Implementación completada

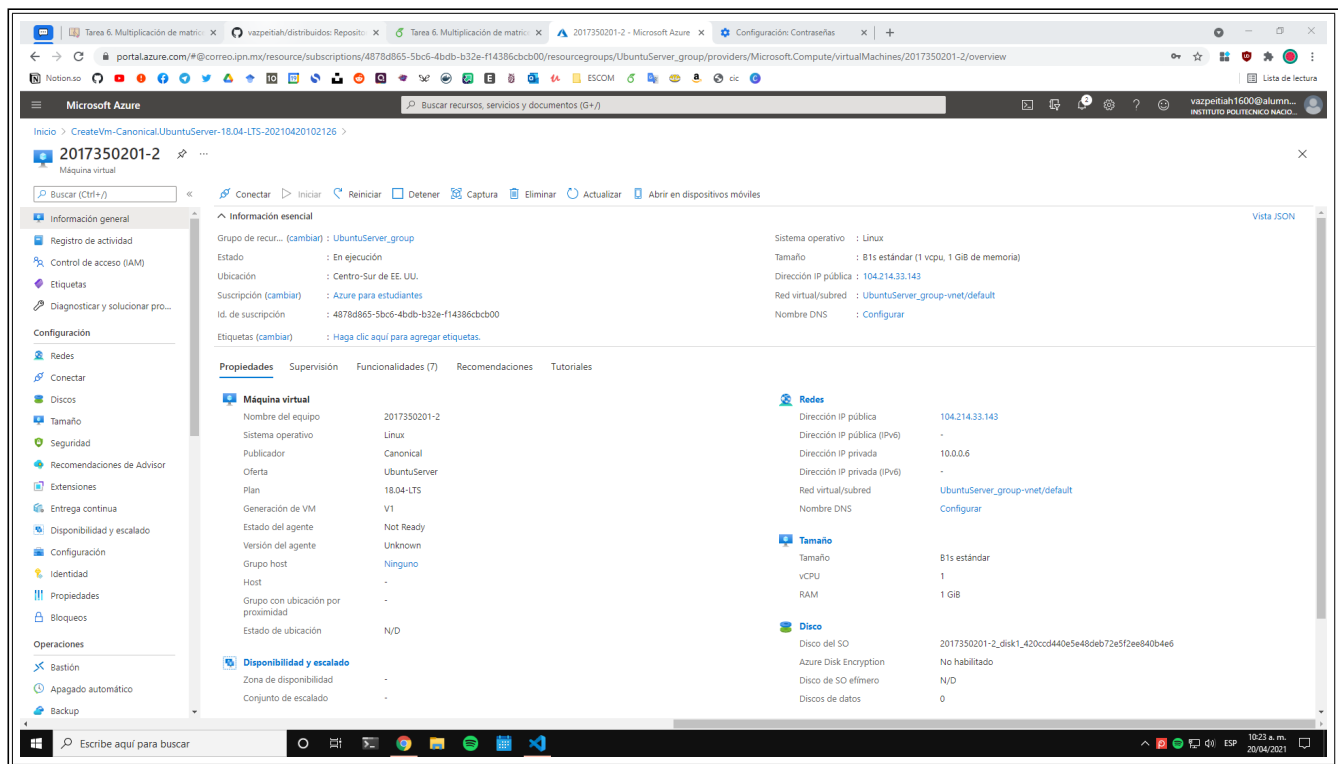


Figura 32: Creación del nodo 2: Panel de control de la máquina virtual

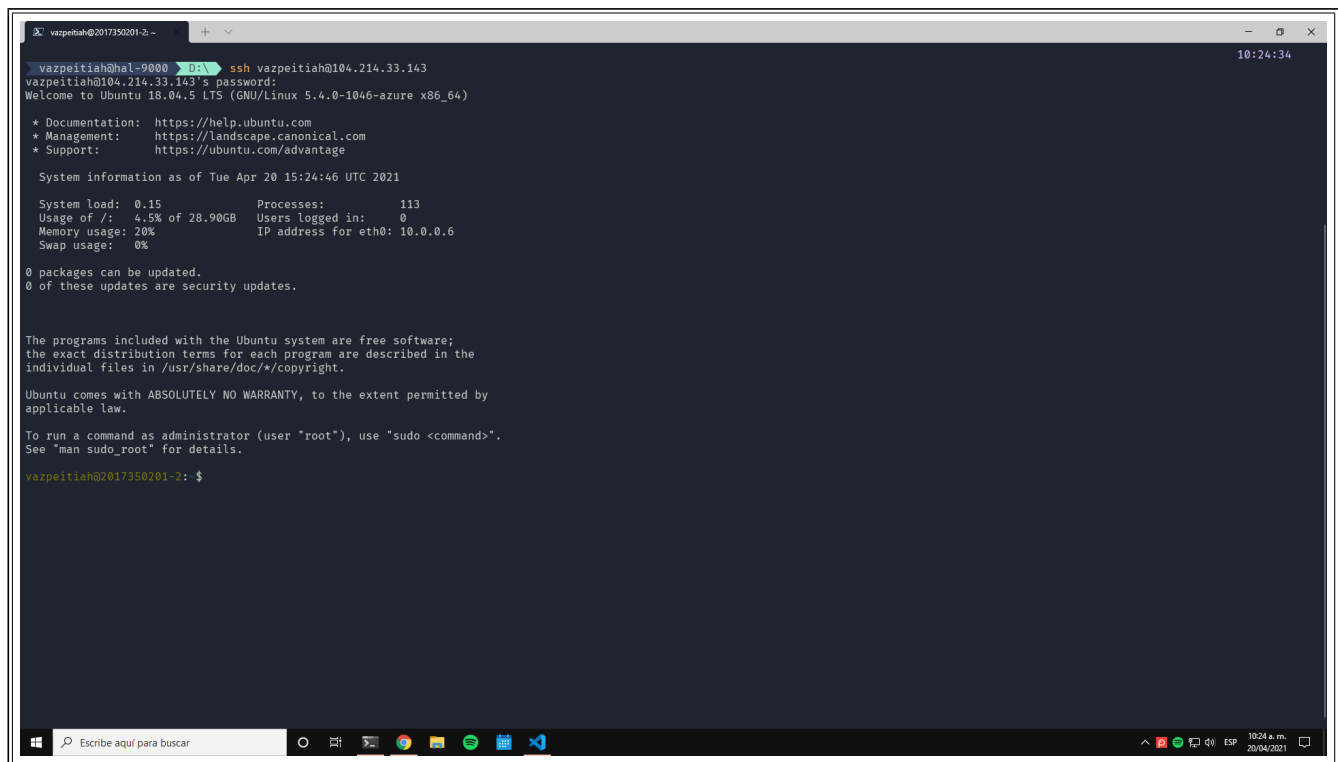


Figura 33: Creación del nodo 2: Conectarse a la máquina virtual a través de ssh

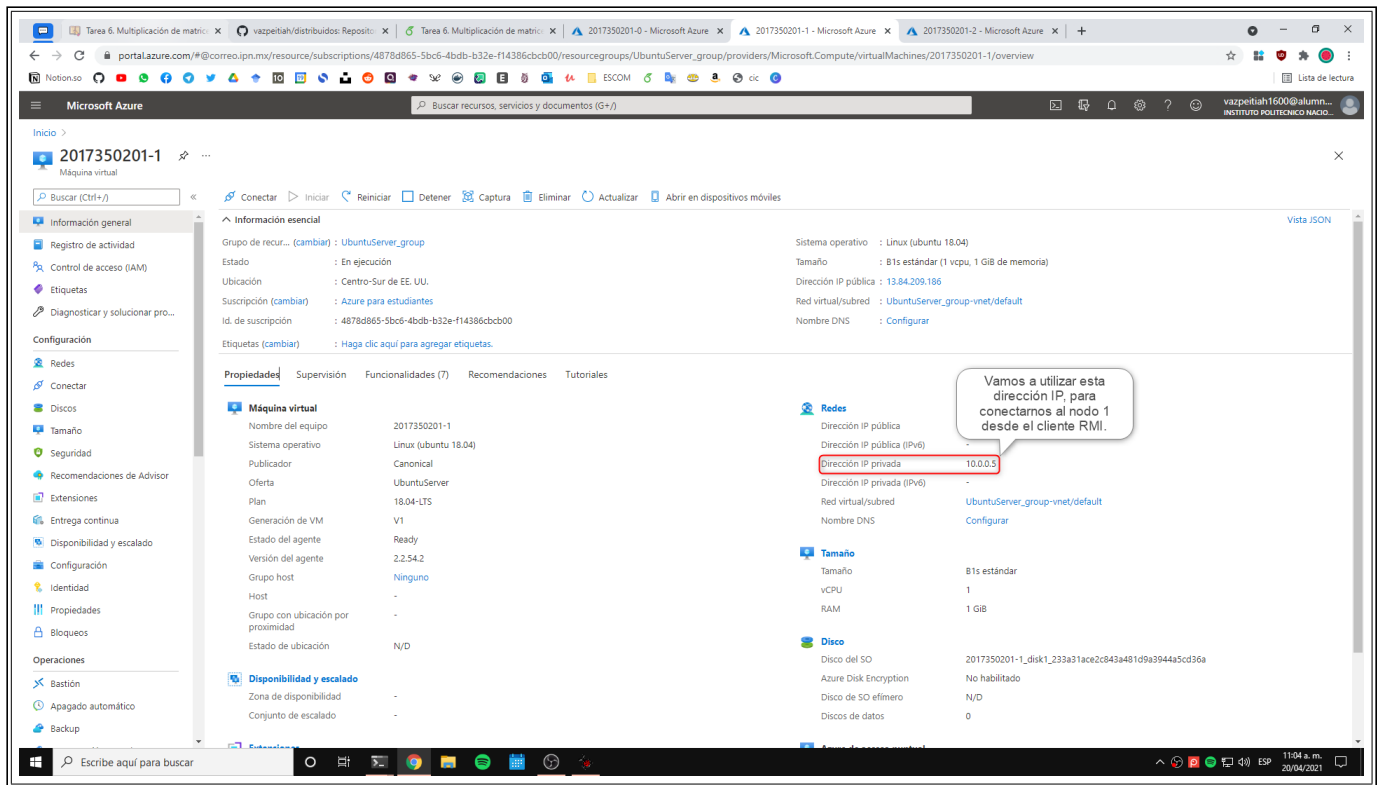


Figura 36: Dirección IP privada del nodo 1

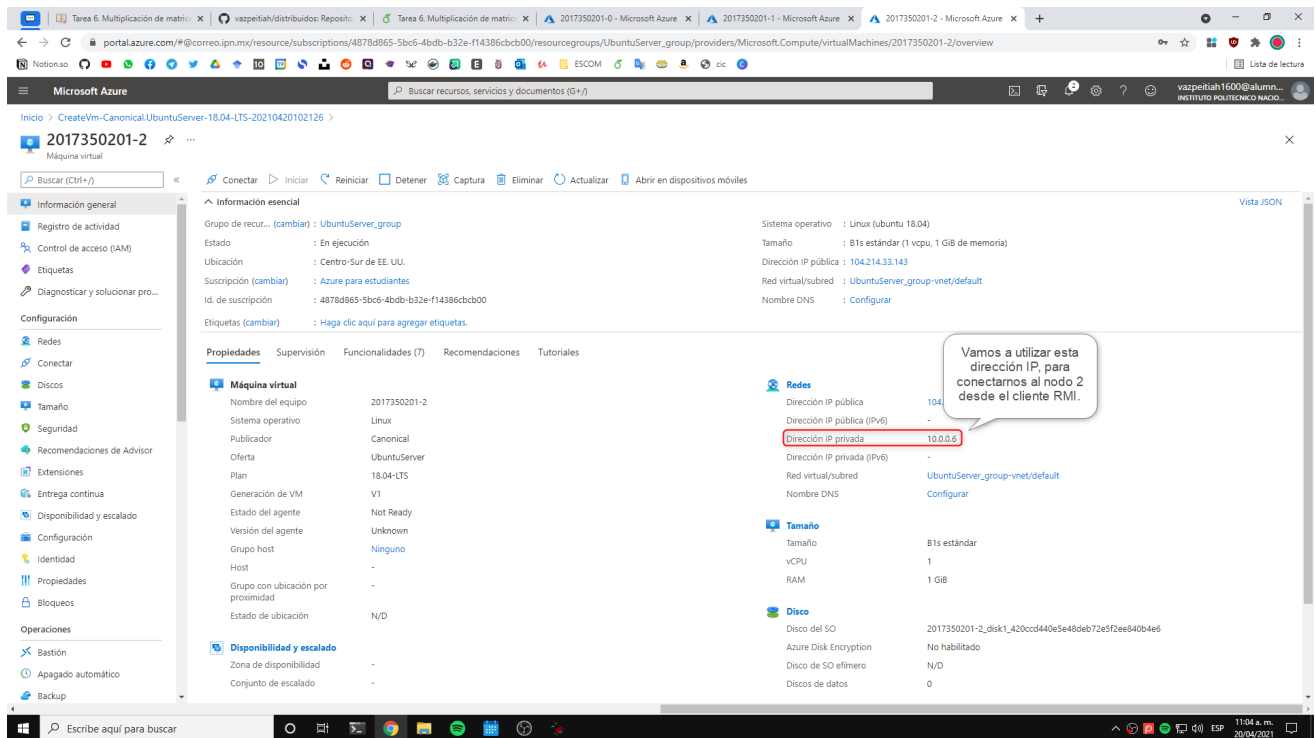


Figura 37: Dirección IP privada del nodo 2

Una vez tengamos las direcciones IP privadas, del nodo 1 y 2, actualizamos el código de la clase ClienteRMI.java, para que accedan al nodo 1 y 2:

```
vazpeitiah@2017350201-0: ~/distribuidos/tarea$  
  
System.out.println("Matriz B:");  
printMatrix(B, N, N);  
  
// transpone la matriz B, la matriz traspuesta queda en B  
for (int i = 0; i < N; i++) {  
    for (int j = 0; j < i; j++) {  
        float x = B[i][j];  
        B[i][j] = B[j][i];  
        B[j][i] = x;  
    }  
}  
  
System.out.println("Matriz B^T:");  
printMatrix(B, N, N);  
  
// Separar la matriz en matrices más pequeñas  
float[][] A1 = separa_matriz(A, 0);  
float[][] A2 = separa_matriz(A, N / 2);  
float[][] B1 = separa_matriz(B, 0);  
float[][] B2 = separa_matriz(B, N / 2);  
  
// obtiene una referencia que "apunta" al objeto remoto asociado a la URL  
//InterfaceRMI nodo1 = (InterfaceRMI) Naming.lookup("rmi://localhost/multmatrix");  
InterfaceRMI nodo1 = (InterfaceRMI) Naming.lookup("rmi://10.0.0.5/multmatrix");  
//InterfaceRMI nodo2 = (InterfaceRMI) Naming.lookup("rmi://localhost/multmatrix");  
InterfaceRMI nodo2 = (InterfaceRMI) Naming.lookup("rmi://10.0.0.6/multmatrix");  
  
// Multiplica las matrices  
float[][] C1 = nodo1.multiplica_matrices(A1, B1);  
float[][] C2 = nodo1.multiplica_matrices(A1, B2);  
float[][] C3 = nodo2.multiplica_matrices(A2, B1);  
float[][] C4 = nodo2.multiplica_matrices(A2, B2);  
  
// Une la matriz resultante  
acomoda_matriz(C, C1, 0, 0);  
acomoda_matriz(C, C2, 0, N / 2);  
acomoda_matriz(C, C3, N / 2, 0);  
acomoda_matriz(C, C4, N / 2, N / 2);  
  
System.out.println("Matriz C:");  
printMatrix(C, N, N);  
  
for (int i = 0; i < N; i++) {  
    for (int j = 0; j < N; j++) {  
        checksum += C[i][j];  
    }  
}
```

Sustituimos "localhost" por la dirección IP privada de cada nodo, respectivamente.

Figura 38: Agregando las direcciones IP privadas, de los nodos 1 y 2, al cliente RMI

```
vazpeitiah@2017350201-0: ~/distribuidos/tarea$ cat Makefile  
JC = javac  
.SUFFIXES: .java .class  
.java.class:  
    $(JC) $*.java  
  
CLASSES = \  
    InterfaceRMI.java \  
    ClaseRMI.java \  
    ClienteRMI.java \  
    ServidorRMI.java  
  
default: classes  
classes: $(CLASSES:.java=.class)  
  
clean:  
    rm *.class  
vazpeitiah@2017350201-0: ~/distribuidos/tarea$ make  
javac InterfaceRMI.java  
javac ClaseRMI.java  
javac ClienteRMI.java  
javac ServidorRMI.java  
vazpeitiah@2017350201-0: ~/distribuidos/tarea$  
  
vazpeitiah@2017350201-1: ~/distribuidos/tarea$ cat Makefile  
JC = javac  
.SUFFIXES: .java .class  
.java.class:  
    $(JC) $*.java  
  
CLASSES = \  
    InterfaceRMI.java \  
    ClaseRMI.java \  
    ClienteRMI.java \  
    ServidorRMI.java  
  
default: classes  
classes: $(CLASSES:.java=.class)  
  
clean:  
    rm *.class  
vazpeitiah@2017350201-1: ~/distribuidos/tarea$ make  
javac InterfaceRMI.java  
javac ClaseRMI.java  
javac ClienteRMI.java  
javac ServidorRMI.java  
vazpeitiah@2017350201-1: ~/distribuidos/tarea$  
  
vazpeitiah@2017350201-2: ~/distribuidos/tarea$ cat Makefile  
JC = javac  
.SUFFIXES: .java .class  
.java.class:  
    $(JC) $*.java  
  
CLASSES = \  
    InterfaceRMI.java \  
    ClaseRMI.java \  
    ClienteRMI.java \  
    ServidorRMI.java  
  
default: classes  
classes: $(CLASSES:.java=.class)  
  
clean:  
    rm *.class  
vazpeitiah@2017350201-2: ~/distribuidos/tarea$ make  
javac InterfaceRMI.java  
javac ClaseRMI.java  
javac ClienteRMI.java  
javac ServidorRMI.java  
vazpeitiah@2017350201-2: ~/distribuidos/tarea$
```

Figura 39: Compilando el código fuente en cada uno de los nodos

Primero debemos correr los dos servidores RMI en el nodo 1 y 2. Antes de ejecutar el servidor RMI debemos ejecutar el comando **rmiregistry &**. El '&' es para dejarlo ejecutándose en segundo plano. Finalmente ejecutamos el cliente RMI, y este nos mostrará los resultados.

Ejecutando el programa para N=8, se imprimen las matrices en consola y se muestra el checksum de C.

```

vazpeitiah@2017350201-0:~/distribuidos/tareas$ make
javac ClienteRMI.java
vazpeitiah@2017350201-1:~/distribuidos/tareas$ clear
vazpeitiah@2017350201-0:~/distribuidos/tareas$ java ClienteRMI
Matriz A:
0.0 3.0 6.0 9.0 12.0 15.0 18.0 21.0
1.0 4.0 7.0 10.0 13.0 16.0 19.0 22.0
2.0 5.0 8.0 11.0 14.0 17.0 20.0 23.0
3.0 6.0 9.0 12.0 15.0 18.0 21.0 24.0
4.0 7.0 10.0 13.0 16.0 19.0 22.0 25.0
5.0 8.0 11.0 14.0 17.0 20.0 23.0 26.0
6.0 9.0 12.0 15.0 18.0 21.0 24.0 27.0
7.0 10.0 13.0 16.0 19.0 22.0 25.0 28.0

Matriz B:
0.0 -2.0 -6.0 -9.0 -12.0 -15.0 -18.0 -21.0
1.0 -2.0 -5.0 -8.0 -11.0 -14.0 -17.0 -20.0
2.0 -1.0 -4.0 -7.0 -10.0 -13.0 -16.0 -19.0
3.0 0.0 -3.0 -6.0 -9.0 -12.0 -15.0 -18.0
4.0 1.0 -2.0 -5.0 -8.0 -11.0 -14.0 -17.0
5.0 2.0 -1.0 -4.0 -7.0 -10.0 -13.0 -16.0
6.0 3.0 0.0 -3.0 -6.0 -9.0 -12.0 -15.0
7.0 4.0 1.0 -2.0 -5.0 -8.0 -11.0 -14.0

Matriz B^T:
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0
-1.0 -2.0 -1.0 0.0 1.0 2.0 3.0 4.0
-6.0 -5.0 -4.0 -3.0 -2.0 -1.0 0.0 1.0
-9.0 -8.0 -7.0 -6.0 -5.0 -4.0 -3.0 -2.0
-12.0 -11.0 -10.0 -9.0 -8.0 -7.0 -6.0 -5.0
-15.0 -14.0 -13.0 -12.0 -11.0 -10.0 -9.0 -8.0
-18.0 -17.0 -16.0 -15.0 -14.0 -13.0 -12.0 -11.0
-21.0 -20.0 -19.0 -18.0 -17.0 -16.0 -15.0 -14.0

Matriz C:
420.0 165.0 -84.0 -336.0 -588.0 -840.0 -1092.0 -1344.0
448.0 172.0 -104.0 -368.0 -656.0 -928.0 -1208.0 -1484.0
476.0 176.0 -124.0 -424.0 -724.0 -1024.0 -1324.0 -1624.0
504.0 188.0 -144.0 -468.0 -792.0 -1116.0 -1440.0 -1764.0
522.0 184.0 -164.0 -512.0 -860.0 -1200.0 -1556.0 -1904.0
560.0 188.0 -184.0 -556.0 -928.0 -1300.0 -1672.0 -2044.0
588.0 192.0 -204.0 -600.0 -996.0 -1392.0 -1788.0 -2184.0
616.0 196.0 -224.0 -644.0 -1064.0 -1484.0 -1904.0 -2324.0
Checksum de C: -42112.0
vazpeitiah@2017350201-0:~/distribuidos/tareas$

vazpeitiah@2017350201-1:~/distribuidos/tareas$ rmiregistry &
[1] 7320
vazpeitiah@2017350201-1:~/distribuidos/tareas$ java ServidorRMI
RMI

vazpeitiah@2017350201-2:~/distribuidos/tareas$ rmiregistry &
[1] 7082
vazpeitiah@2017350201-2:~/distribuidos/tareas$ java ServidorRMI
RMI
```

Figura 40: Ejecutando el programa para N=8

Ejecutando el programa para N=1000, solo se muestra el checksum de la matriz C.

```

vazpeitiah@2017350201-0:~/distribuidos/tareas$ java ClienteRMI
Checksum de C: -1.746002227639148E15
vazpeitiah@2017350201-0:~/distribuidos/tareas$

vazpeitiah@2017350201-1:~/distribuidos/tareas$ java ServidorRMI
RMI

vazpeitiah@2017350201-2:~/distribuidos/tareas$ java ServidorRMI
RMI
```

Figura 41: Ejecutando el programa para N=1000

3. Conclusiones

En esta práctica implementamos un programa similar al de la tarea 3, pero ahora utilizamos RMI. Es más sencillo hacerlo de esta manera, ya que no debemos preocuparnos por el envío de las matrices. Para hacer las pruebas creamos 3 máquinas virtuales de Ubuntu Server, en Microsoft Azure. En este caso la implementación de las máquinas es muy sencilla y la conexión a través de ssh facilita mucho las cosas. Me pareció interesante esta forma de usar el cómputo distribuido, por su facilidad y agilidad a la hora de implementar un programa con estas características.