



INSTITUTO POLITÉCNICO NACIONAL  
ESCUELA SUPERIOR DE CÓMPUTO

## Tarea 3. Multiplicación de matrices distribuida utilizando paso de mensajes

Unidad de aprendizaje: Desarrollo de Sistemas Distribuidos

Grupo: 4CV11

*Alumno:*  
Sanchez Mendez Edmundo Josue

*Profesor:*  
Pineda Guerrero Carlos

13 de octubre de 2021

# Índice

<b>1. Introducción</b>	<b>2</b>
<b>2. Desarrollo</b>	<b>3</b>
2.1. Compilación del código . . . . .	8
2.2. Ejecución del programa . . . . .	9
2.2.1. N=10 . . . . .	9
2.2.2. N=1500 . . . . .	11
2.3. Código . . . . .	13
<b>3. Conclusiones</b>	<b>17</b>

## 1. Introducción

Mediante el desarrollo de un sistema distribuido el cual tendrá una topología lógica de estrella en donde interceden 5 nodos, en el cual el nodo 0 sera el nodo central de la topología, el objetivo de la practica es llevar acabo una multiplicación de matrices de tamaño  $N$ , para esta practica  $N$  sera igual a 10 y a 1500.

El funcionamiento del programa es el siguiente:

- Se tendrán 3 matrices, A, B y C de tipo long y tamaño  $N \times N$ .
- El nodo 0 sera el encargado de realizar lo siguiente:
  - Inicializar las matrices A y B con los siguientes valores:  $A[i][j] = i + 2 * j$  y  $B[i][j] = i - 2 * j$ .
  - Transponer la matriz B.
  - Enviar la matriz A1 al nodo 1.
  - Enviar la matriz B1 al nodo 1.
  - Enviar la matriz A1 al nodo 2.
  - Enviar la matriz B2 al nodo 2.
  - Enviar la matriz A2 al nodo 3.
  - Enviar la matriz B1 al nodo 3.
  - Enviar la matriz A2 al nodo 4.
  - Enviar la matriz B2 al nodo 4.
  - Recibir la matriz C1 del nodo 1.
  - Recibir la matriz C2 del nodo 2.
  - Recibir la matriz C3 del nodo 3.
  - Recibir la matriz C4 del nodo 4.
  - Calcular el checksum de la matriz C, con base a la siguiente formula:

$$\sum_{i=0, j=0}^{N-1} C[i][j]$$

- 
- Desplegar el checksum de la matriz C.
- Si  $N=10$  entonces desplegar las matrices A, B y C
- El nodo 1 sera el encargado de realizar lo siguiente.
  - Recibir del nodo 0 la matriz A1.
  - Recibir del nodo 0 la matriz B1.
  - Realizar el producto  $C1 = A1 \times B1$  (renglón por renglón).
  - Enviar la matriz C1 al nodo 0.
- El nodo 2 sera el encargado de realizar lo siguiente.
  - Recibir del nodo 0 la matriz A1.
  - Recibir del nodo 0 la matriz B2.

- Realizar el producto  $C2=A1 \times B2$  (renglón por renglón).
  - Enviar la matriz  $C2$  al nodo 0.
- El nodo 3 sera el encargado de realizar lo siguiente.
- Recibir del nodo 0 la matriz  $A2$ .
  - Recibir del nodo 0 la matriz  $B1$ .
  - Realizar el producto  $C3=A2 \times B1$  (renglón por renglón).
  - Enviar la matriz  $C3$  al nodo 0.
- El nodo 4 sera el encargado de realizar lo siguiente.
- Recibir del nodo 0 la matriz  $A2$ .
  - Recibir del nodo 0 la matriz  $B2$ .
  - Realizar el producto  $C4=A2 \times B2$  (renglón por renglón).
  - Enviar la matriz  $C4$  al nodo 0.

## 2. Desarrollo

Para poder desarrollar esta tarea se tomo como ejemplo la tarea 1 y el código enseñado en clase de multiplicación de matrices, basándonos en el archivo `MultiplicaMatriz_2.java`. Una vez programado el sistema distribuido con base en los requerimientos solicitados para la tarea se procede a crear las 5 maquinas virtuales en Azure para el cumplimiento de el tipo de topologia que estamos usando. En las figuras de la 1 a la 7 se ve la creación del nodo 0 como maquina virtual en Azure cumpliendo con las características señaladas en la tarea la cual se nos pide que el OS sea Ubuntu con 1 CPU, 1GB de RAM y disco HDD estándar, ademas de llevar como nombre `M2019630428-n`, donde n es el número de nodo.

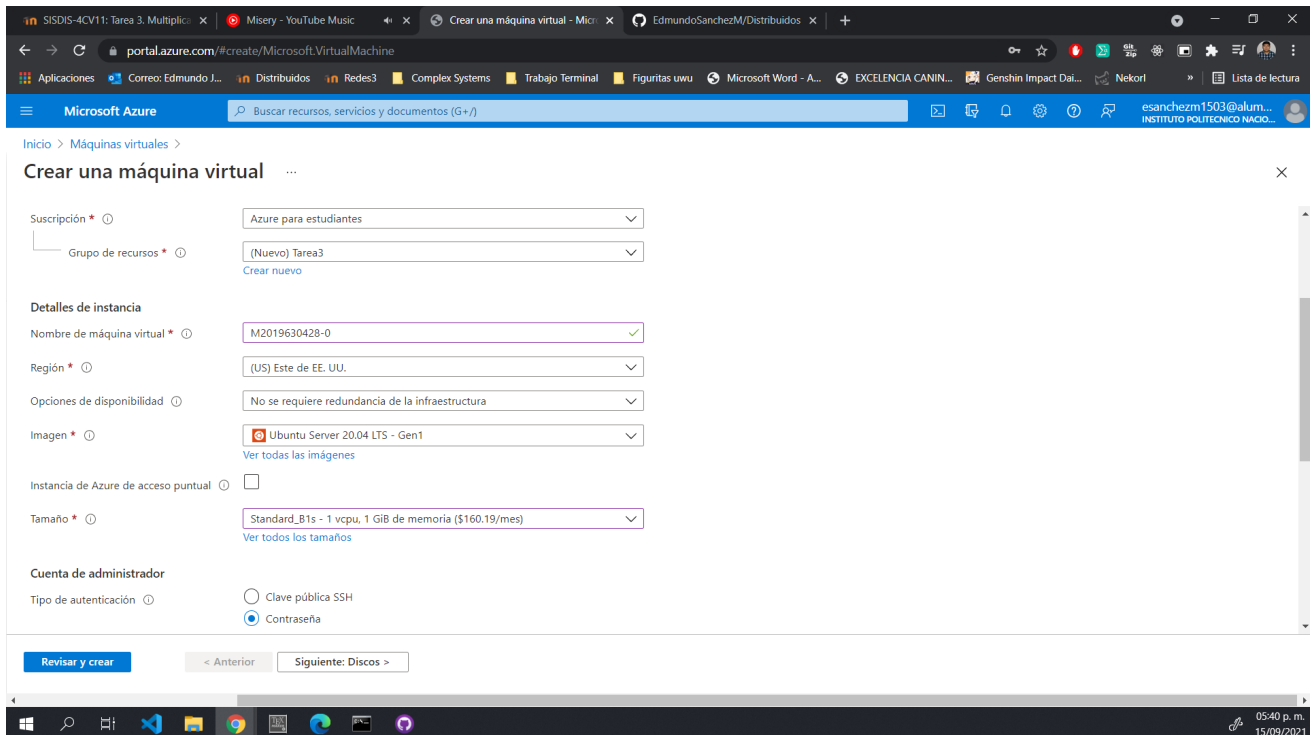


Figura 1: Datos básicos para el nodo 0.

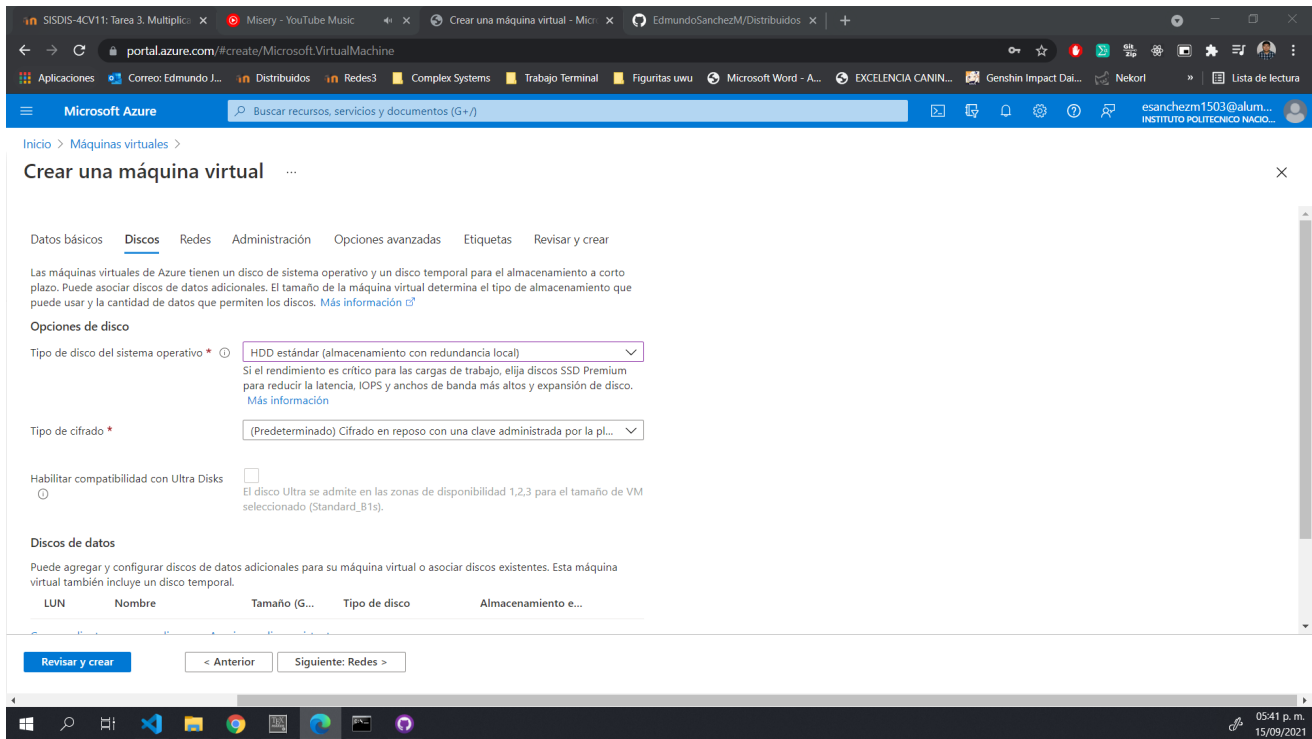


Figura 2: Configuración del tipo de disco para el nodo 0.

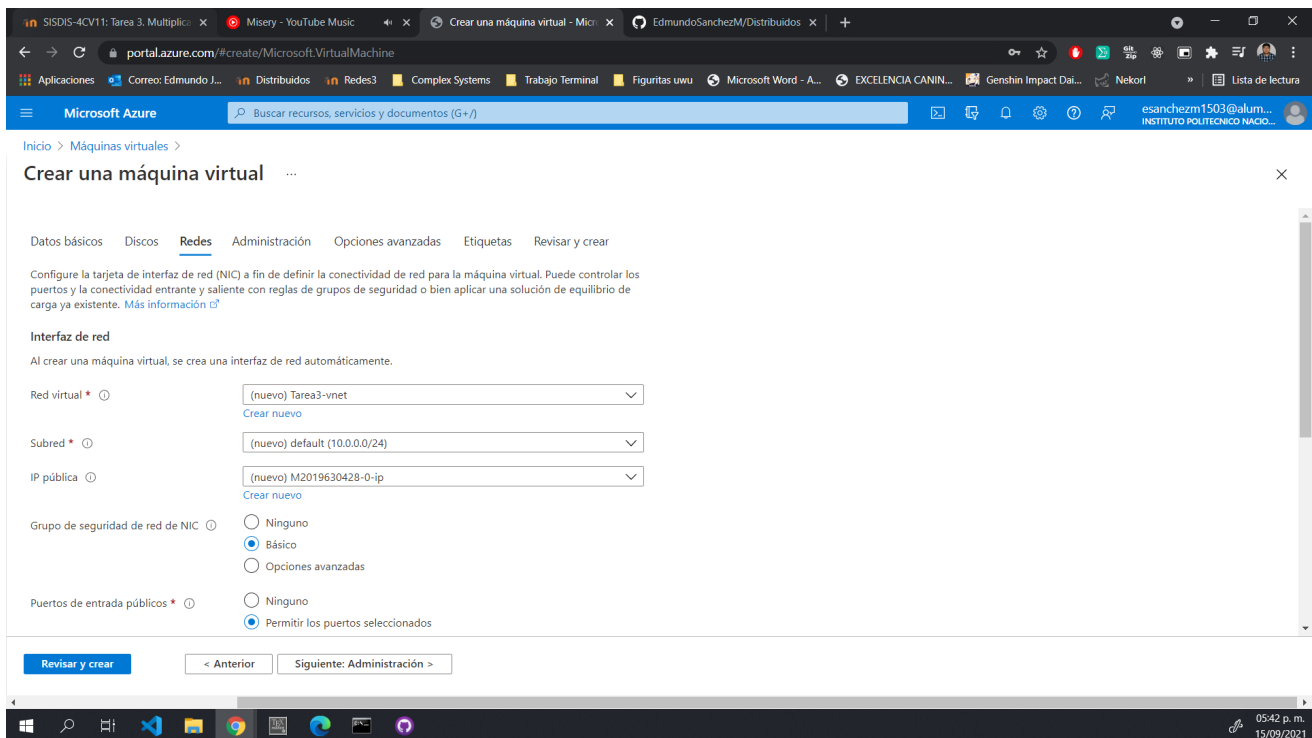


Figura 3: Información sobre la redes para el nodo 0.

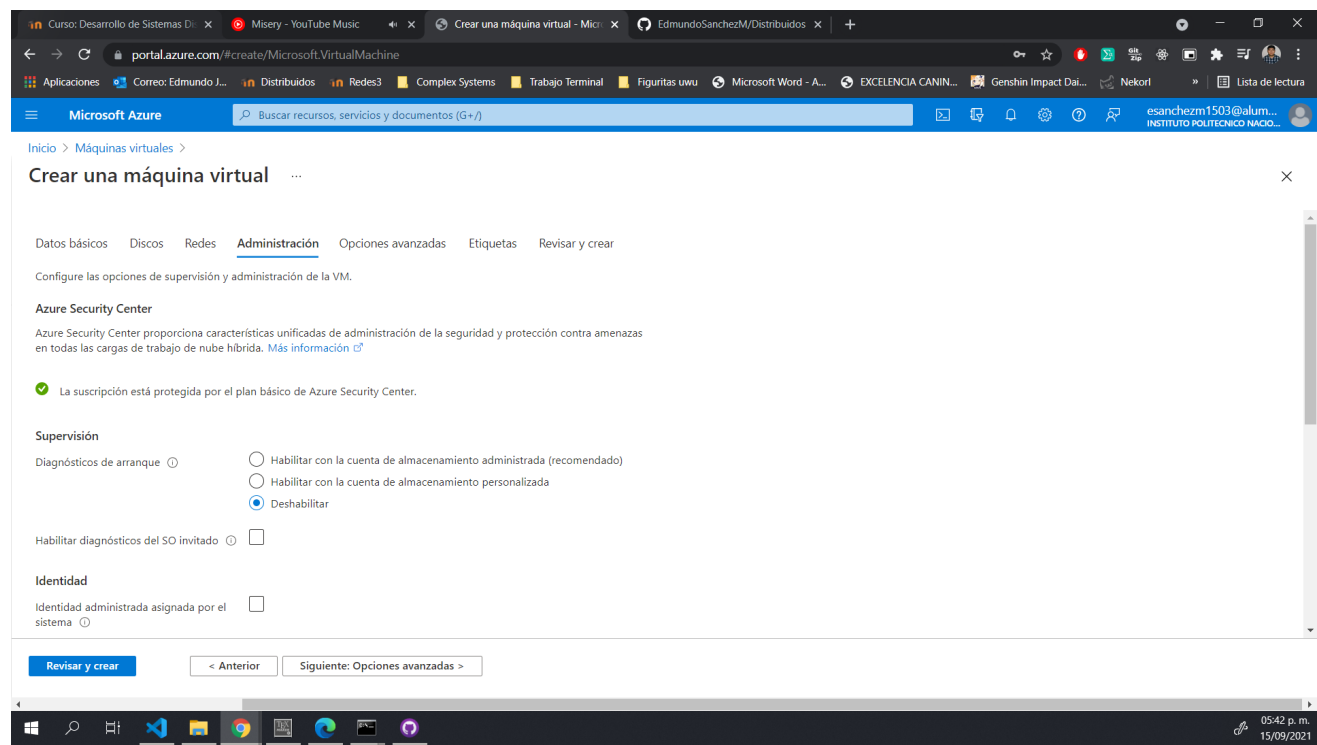


Figura 4: Configuración de la administración para el nodo 0.

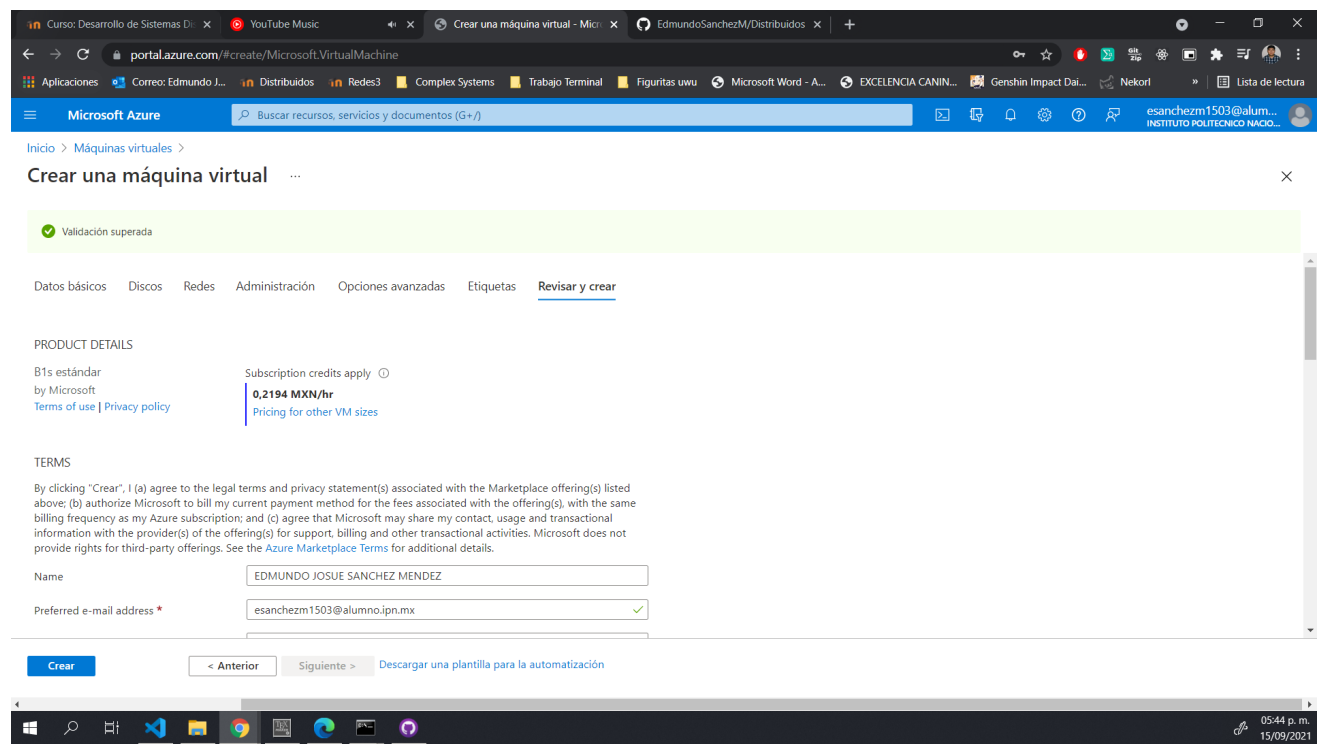


Figura 5: Creación de la maquina virtual para el nodo 0.

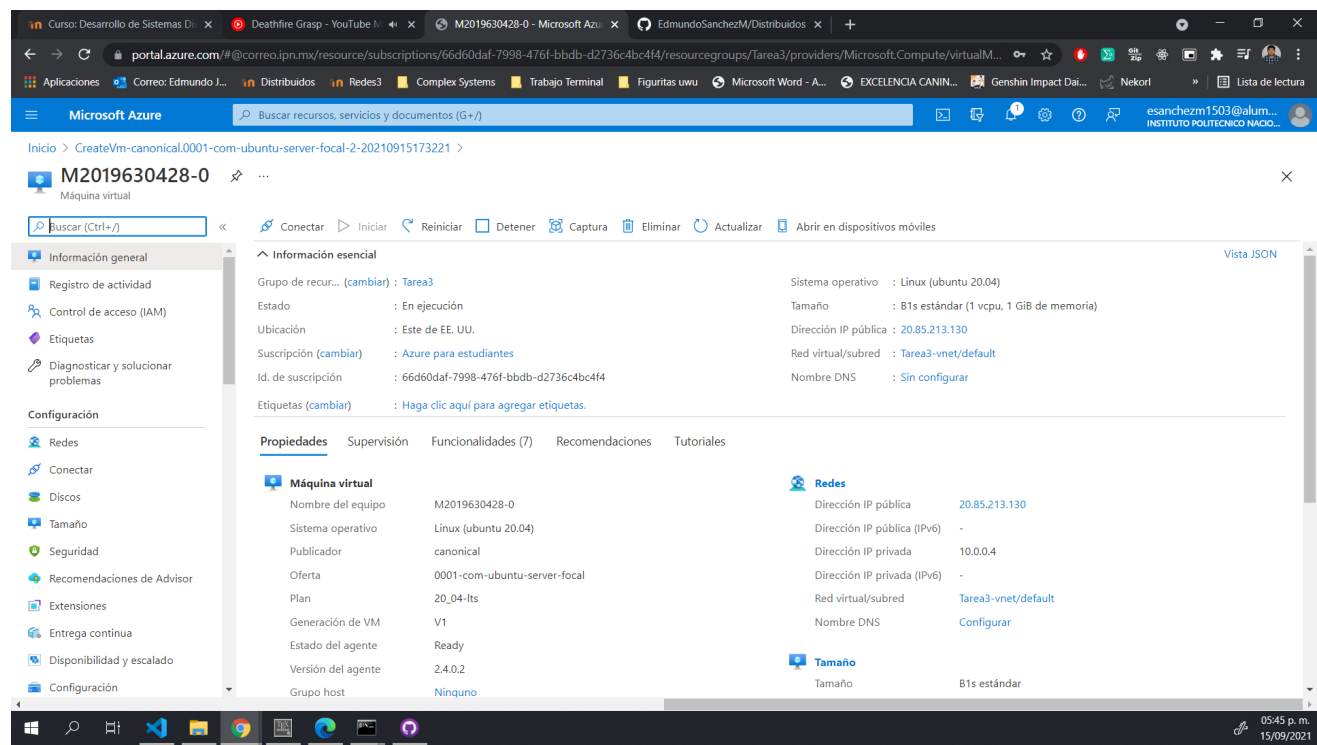


Figura 6: Panel de control de la maquina virtual para el nodo 0.

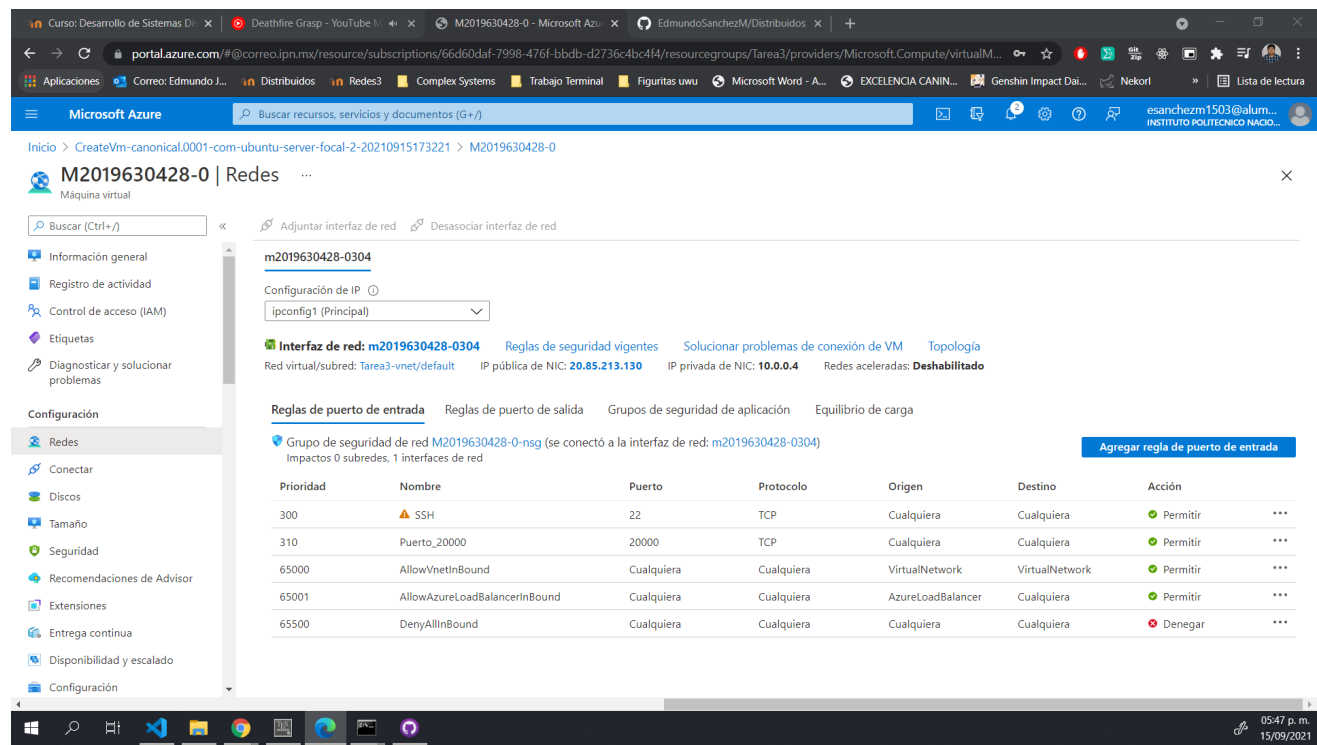
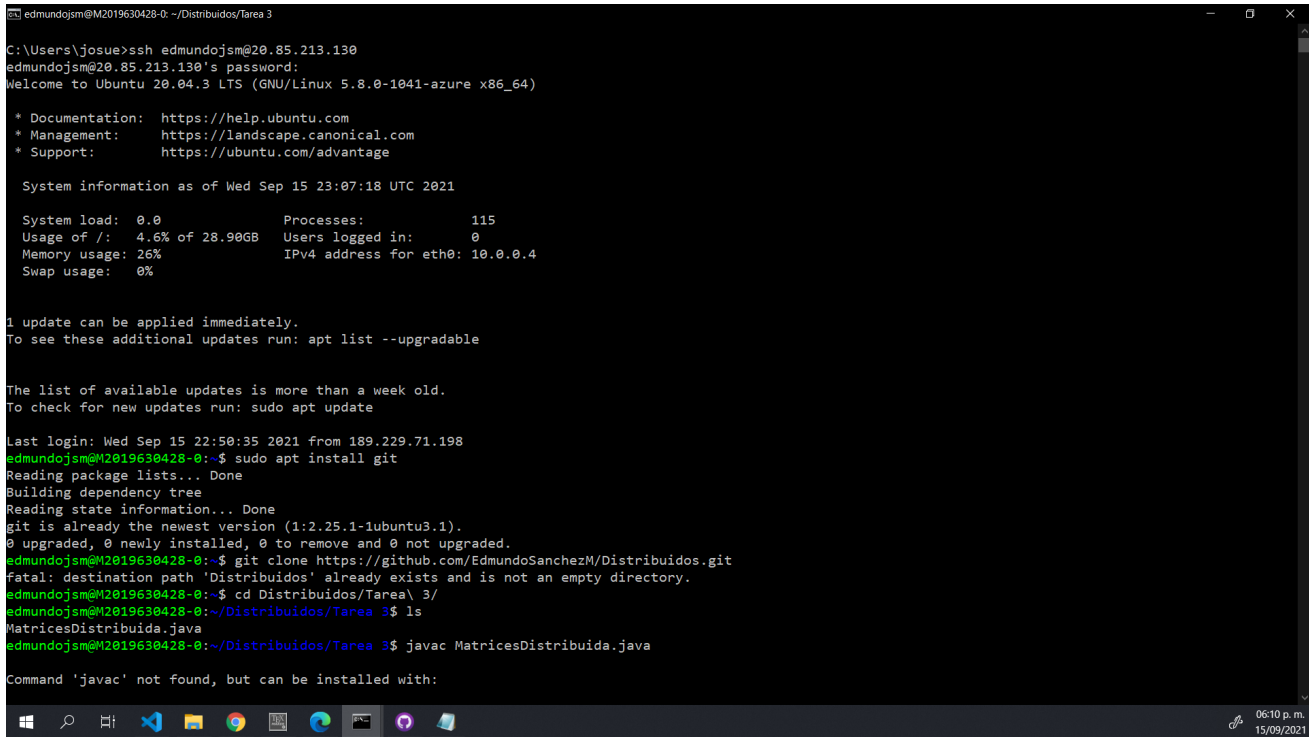


Figura 7: Apertura del puerto 20000 en el nodo 0.

Las imágenes de la 1 a la 7 se tuvieron que repetir cuatro veces mas, esto debido al tipo de topologia a implementar, mencionar que en mi caso al momento de querer crear una quinta maquina virtual con las características dadas en la tarea no se me dejo, ya que tengo un limite de 4 maquinas al usar standar.

B1, por lo que tuve que crear una maquina virtual con otras características, intente aumentar el limite que tengo pero no me fue posible ya que me solicitan crear un token con el soporte técnico, sin embargo, al usar una maquina con otras características aparentemente no me genero un gran costo como pensaba que seria.

Una vez creadas las maquinas virtuales se procedió a hacer lo que se muestra en las figuras de 8 y 9 por las 5 maquinas que tenemos, usamos como ejemplo el nodo 0



```
edmundojasm@M2019630428-0: ~/Distribuidos/Tarea 3
C:\Users\josue>ssh edmundojasm@20.85.213.130
edmundojasm@20.85.213.130's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.8.0-1041-azure x86_64)

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

System information as of Wed Sep 15 23:07:18 UTC 2021

System load:  0.0          Processes:    115
Usage of /:   4.6% of 28.9GB Users logged in:   0
Memory usage: 26%         IPv4 address for eth0: 10.0.0.4
Swap usage:   0%

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed Sep 15 22:50:35 2021 from 189.229.71.198
edmundojasm@M2019630428-0:~$ sudo apt install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.25.1-1ubuntu3.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
edmundojasm@M2019630428-0:~$ git clone https://github.com/EdmundoSanchezM/Distribuidos.git
fatal: destination path 'Distribuidos' already exists and is not an empty directory.
edmundojasm@M2019630428-0:~$ cd Distribuidos/Tarea 3/
edmundojasm@M2019630428-0:~/Distribuidos/Tarea 3$ ls
MatricesDistribuida.java
edmundojasm@M2019630428-0:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
Command 'javac' not found, but can be installed with:
```

Figura 8: Conexión vía ssh a la maquina virtual que hará como nodo 0.



```

edmundojm@M2019630428-0: ~/Distribuidos/Tarea 3
sudo apt install openjdk-16-jdk-headless # version 16.0.1+9-1~20.04
sudo apt install openjdk-8-jdk-headless # version 8u292-b10-0ubuntu1~20.04
sudo apt install openjdk-13-jdk-headless # version 13.0.7+5-0ubuntu1~20.04
sudo apt install ecj # version 3.16.0-1

edmundojm@M2019630428-0:~/Distribuidos/Tarea 3$ sudo apt install default-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  default-jdk-headless libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk
  openjdk-11-jdk-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source visualvm
The following NEW packages will be installed:
  default-jdk default-jdk-headless libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk
  openjdk-11-jdk-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 16 newly installed, 0 to remove and 0 not upgraded.
Need to get 226 MB of archives.
After this operation, 242 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://azure.archive.ubuntu.com/ubuntu focal-updates/main amd64 openjdk-11-jdk-headless amd64 11.0.11+9-0ubuntu2~20.04 [223 MB]
Get:2 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 default-jdk-headless amd64 2:1.11-72 [1140 B]
Get:3 http://azure.archive.ubuntu.com/ubuntu focal-updates/main amd64 openjdk-11-jdk amd64 11.0.11+9-0ubuntu2~20.04 [1442 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 default-jdk amd64 2:1.11-72 [1096 B]
Get:5 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 xorg-sgml-doctools all 1:1.11-1 [12.9 kB]
Get:6 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 x11proto-dev all 2019.2-1ubuntu1 [594 kB]
Get:7 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 x11proto-core-dev all 2019.2-1ubuntu1 [2620 B]
Get:8 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libice-dev amd64 2:1.0.10-0ubuntu1 [47.8 kB]
Get:9 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libpthread-stubs0-dev amd64 0.4-1 [5384 B]
Get:10 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libsm-dev amd64 2:1.2.3-1 [17.0 kB]
Get:11 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libxau-dev amd64 1:1.0.9-0ubuntu1 [9552 B]
Get:12 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libxdmcp-dev amd64 1:1.1.3-0ubuntu1 [25.3 kB]
Get:13 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 xtrans-dev all 1.4.0-1 [68.9 kB]
Get:14 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libxcb1-dev amd64 1.14-2 [80.5 kB]
Get:15 http://azure.archive.ubuntu.com/ubuntu focal-updates/main amd64 libx11-dev amd64 2:1.6.9-2ubuntu1.2 [647 kB]
Get:16 http://azure.archive.ubuntu.com/ubuntu focal/main amd64 libxt-dev amd64 1:1.1.5-1 [395 kB]
Fetched 226 MB in 3s (74.1 MB/s)

```

Figura 9: Instalación del jdk en el nodo 0.

Una vez configurado cada maquina virtual se procedió a clonar un repositorio privado de mi autoría almacenado en GitHub, esto para poder compartir el archivo necesario para el funcionamiento de la practica, esto se puede ver en la figura 11. Una vez de tener configurada la maquina virtual y teniendo el archivo necesario para el cumplimiento de la tarea se procede a la siguiente fase de la tarea, la compilación del código.

## 2.1. Compilación del código

En la figura 10 podemos ver como la compilación de nuestro programa `MatricesDistribuida.java` se hace de manera exitosa y sin ningún error alguno, esto desde la terminal de Ubuntu de nuestra maquina virtual.

```

edmundojm@M2019630428-0: ~/Distribuidos/Tarea 3
Setting up x11proto-dev (2019.2-1ubuntu1) ...
Setting up libxau-dev:amd64 (1:1.0.9-0ubuntu1) ...
Setting up libice-dev:amd64 (2:1.0.10-0ubuntu1) ...
Setting up libsm-dev:amd64 (2:1.2.3-1) ...
Setting up default-jdk (2:1.11-72) ...
Setting up libxdmcp-dev:amd64 (1:1.1.3-0ubuntu1) ...
Setting up x11proto-core-dev (2019.2-1ubuntu1) ...
Setting up libxcb1-dev:amd64 (1.14-2) ...
Setting up libx11-dev:amd64 (2:1.6.9-2ubuntu1.2) ...
Setting up libxt-dev:amd64 (1:1.1.5-1) ...
Processing triggers for man-db (2.9.1-1) ...
edmundojm@M2019630428-0:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojm@M2019630428-0:~/Distribuidos/Tarea 3$ java MatricesDistribuida 0 20.85.213.130

Valor Matriz A
0 2 4 6 8 10 12 14 16 18
1 3 5 7 9 11 13 15 17 19
2 4 6 8 10 12 14 16 18 20
3 5 7 9 11 13 15 17 19 21
4 6 8 10 12 14 16 18 20 22
5 7 9 11 13 15 17 19 21 23
6 8 10 12 14 16 18 20 22 24
7 9 11 13 15 17 19 21 23 25
8 10 12 14 16 18 20 22 24 26
9 11 13 15 17 19 21 23 25 27

Valor Matriz B
0 -2 -4 -6 -8 -10 -12 -14 -16 -18
1 -1 -3 -5 -7 -9 -11 -13 -15 -17
2 0 -2 -4 -6 -8 -10 -12 -14 -16
3 1 -1 -3 -5 -7 -9 -11 -13 -15
4 2 0 -2 -4 -6 -8 -10 -12 -14
5 3 1 -1 -3 -5 -7 -9 -11 -13
6 4 2 0 -2 -4 -6 -8 -10 -12
7 5 3 1 -1 -3 -5 -7 -9 -11
8 6 4 2 0 -2 -4 -6 -8 -10
9 7 5 3 1 -1 -3 -5 -7 -9

```

Figura 10: Compilación del código por medio de la terminal de Ubuntu de nuestra maquina virtual como nodo 0.

## 2.2. Ejecución del programa

### 2.2.1. N=10

Una vez compilado nuestro programa en cada maquina virtual se procede a la ejecución de los nodos clientes como se ve en la figura 11 y del nodo servidor ver la figura 10.

```

edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3
Password for 'https://EdmundoSanchezM@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 0), reused 4 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), 3.14 KiB | 292.00 KiB/s, done.
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$ cd Distribuidos/Tarea\ 3/
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$ java MatricesDistribuida 1 20
.85.213.130
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$

edmundojsm@M2019630428-2:~/Distribuidos/Tarea 3
Username for 'https://github.com': EdmundoSanchezM
Password for 'https://EdmundoSanchezM@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 0), reused 4 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), 3.14 KiB | 322.00 KiB/s, done.
edmundojsm@M2019630428-2:~/Distribuidos/Tarea 3$ cd Distribuidos/Tarea\ 3/
edmundojsm@M2019630428-2:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojsm@M2019630428-2:~/Distribuidos/Tarea 3$ java MatricesDistribuida 2 20
.85.213.130
edmundojsm@M2019630428-2:~/Distribuidos/Tarea 3$

edmundojsm@M2019630428-3:~/Distribuidos/Tarea 3
git is already the newest version (1:2.25.1-1ubuntu3.1).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
edmundojsm@M2019630428-3:~/Distribuidos/Tarea 3$ git clone https://github.com/EdmundoSanchezM/Distribuidos.git
Cloning into 'Distribuidos'...
Username for 'https://github.com': EdmundoSanchezM
Password for 'https://EdmundoSanchezM@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 0), reused 4 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), 3.14 KiB | 292.00 KiB/s, done.
edmundojsm@M2019630428-3:~/Distribuidos/Tarea 3$ cd Distribuidos/Tarea\ 3/
edmundojsm@M2019630428-3:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojsm@M2019630428-3:~/Distribuidos/Tarea 3$ java MatricesDistribuida 3 20
.85.213.130
edmundojsm@M2019630428-3:~/Distribuidos/Tarea 3$

edmundojsm@M2019630428-4:~/Distribuidos/Tarea 3
git is already the newest version (1:2.25.1-1ubuntu3.1).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
edmundojsm@M2019630428-4:~/Distribuidos/Tarea 3$ git clone https://github.com/EdmundoSanchezM/Distribuidos.git
Cloning into 'Distribuidos'...
Username for 'https://github.com': EdmundoSanchezM
Password for 'https://EdmundoSanchezM@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 0), reused 4 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), 3.14 KiB | 402.00 KiB/s, done.
edmundojsm@M2019630428-4:~/Distribuidos/Tarea 3$ cd Distribuidos/Tarea\ 3/
edmundojsm@M2019630428-4:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojsm@M2019630428-4:~/Distribuidos/Tarea 3$ java MatricesDistribuida 4 20
.85.213.130
edmundojsm@M2019630428-4:~/Distribuidos/Tarea 3$

```

Figura 11: Ejecución del programa en los 4 nodos clientes.

Al finalizar los 4 nodos clientes podemos ver como en el nodo servidor, es decir, el nodo 0 se despliega la matriz C y el valor de checksum de ésta (ver imagen 12 y 13).

```

edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ java MatricesDistribuida 0 20.85.213.130

Valor Matriz A
0 2 4 6 8 10 12 14 16 18
1 3 5 7 9 11 13 15 17 19
2 4 6 8 10 12 14 16 18 20
3 5 7 9 11 13 15 17 19 21
4 6 8 10 12 14 16 18 20 22
5 7 9 11 13 15 17 19 21 23
6 8 10 12 14 16 18 20 22 24
7 9 11 13 15 17 19 21 23 25
8 10 12 14 16 18 20 22 24 26
9 11 13 15 17 19 21 23 25 27

Valor Matriz B
0 -2 -4 -6 -8 -10 -12 -14 -16 -18
1 -1 -3 -5 -7 -9 -11 -13 -15 -17
2 0 -2 -4 -6 -8 -10 -12 -14 -16
3 1 -1 -3 -5 -7 -9 -11 -13 -15
4 2 0 -2 -4 -6 -8 -10 -12 -14
5 3 1 -1 -3 -5 -7 -9 -11 -13
6 4 2 0 -2 -4 -6 -8 -10 -12
7 5 3 1 -1 -3 -5 -7 -9 -11
8 6 4 2 0 -2 -4 -6 -8 -10
9 7 5 3 1 -1 -3 -5 -7 -9

Valor Matriz C
570 390 210 30 -150 -330 -510 -690 -870 -1050
615 415 215 15 -185 -385 -585 -785 -985 -1185
660 440 220 0 -220 -440 -660 -880 -1100 -1320
705 465 225 -15 -255 -495 -735 -975 -1215 -1455
750 490 230 -30 -290 -550 -810 -1070 -1330 -1590
795 515 235 -45 -325 -605 -885 -1165 -1445 -1725
840 540 240 -60 -360 -660 -960 -1260 -1560 -1860
885 565 245 -75 -395 -715 -1035 -1355 -1675 -1995
930 590 250 -90 -430 -770 -1110 -1450 -1790 -2130
975 615 255 -105 -465 -825 -1185 -1545 -1905 -2265

```

Figura 12: Final de la ejecución del programa en el nodo 0. Parte 1.

```

edmundojm@M2019630428-0: ~/Distribuidos/Tarea 3
2   4   6   8   10  12  14  16  18  20
3   5   7   9   11  13  15  17  19  21
4   6   8   10  12  14  16  18  20  22
5   7   9   11  13  15  17  19  21  23
6   8   10  12  14  16  18  20  22  24
7   9   11  13  15  17  19  21  23  25
8   10  12  14  16  18  20  22  24  26
9   11  13  15  17  19  21  23  25  27

Valor Matriz B
0   -2   -4   -6   -8   -10  -12  -14  -16  -18
1   -1   -3   -5   -7   -9   -11  -13  -15  -17
2   0   -2   -4   -6   -8   -10  -12  -14  -16
3   1   -1   -3   -5   -7   -9   -11  -13  -15
4   2   0   -2   -4   -6   -8   -10  -12  -14
5   3   1   -1   -3   -5   -7   -9   -11  -13
6   4   2   0   -2   -4   -6   -8   -10  -12
7   5   3   1   -1   -3   -5   -7   -9   -11
8   6   4   2   0   -2   -4   -6   -8   -10
9   7   5   3   1   -1   -3   -5   -7   -9

Valor Matriz C
570  390  210  30   -150  -330  -510  -690  -870  -1050
615  415  215  15   -185  -385  -585  -785  -985  -1185
660  440  220  0    -220  -440  -660  -880  -1100  -1320
705  465  225  -15  -255  -495  -735  -975  -1215  -1455
750  490  230  -30  -290  -550  -810  -1070  -1330  -1590
795  515  235  -45  -325  -605  -885  -1165  -1445  -1725
840  540  240  -60  -360  -660  -960  -1260  -1560  -1860
885  565  245  -75  -395  -715  -1035  -1355  -1675  -1995
930  590  250  -90  -430  -770  -1110  -1450  -1790  -2130
975  615  255  -105  -465  -825  -1185  -1545  -1905  -2265

Checksum: -44250
edmundojm@M2019630428-0:~/Distribuidos/Tarea 3$

```

Figura 13: Final de la ejecución del programa en el nodo 0. Parte 2.

Como vemos se nos despliega el los valores de la matriz A, B y C así como el checksum de la matriz C, que es como se nos pide en la tarea. Veamos ahora el siguiente caso con  $N=1500$ .

### 2.2.2. $N=1500$

Para este caso se tuvo que modificar el programa asignando 1500 a la variable N, volver a subir el archivo a GitHub y usar el comando git pull para recibir las actualizaciones del repositorio, volver a compilar el archivo y ejecutarlo, todo lo anterior se puede ver en la figura 14.

The figure shows four terminal windows, each representing a client node (1, 2, 3, and 4). Each window displays the following sequence of commands and output:

```

edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$ git pull
Username for 'https://github.com': EdmundoSanchezM
Password for 'https://EdmundoSanchezM@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 1), reused 4 (delta 1), pack-reused 0
Unpacking objects: 100% (4/4), 374 bytes | 374.00 KiB/s, done.
From https://github.com/EdmundoSanchezM/Distribuidos
   9a88c26..567b980  main       -> origin/main
Updating 9a88c26..567b980
Fast-forward
   Tarea 3/MatricesDistribuida.java | 2 +-
   1 file changed, 1 insertion(+), 1 deletion(-)
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$ java MatricesDistribuida 1 20
.85.213.130
edmundojsm@M2019630428-1:~/Distribuidos/Tarea 3$

```

The same sequence of commands and output is repeated for nodes 2, 3, and 4, with the final command being `java MatricesDistribuida 2 20`, `java MatricesDistribuida 3 20`, and `java MatricesDistribuida 4 20` respectively.

Figura 14: Actualización, compilación y ejecución del programa en los 4 nodos clientes.

Al finalizar los 4 nodos clientes podemos ver como en el nodo servidor, es decir, el nodo 0 solo se nos despliega el valor del checksum de la matriz C obtenida, al mismo tiempo podemos ver la actualización, compilación y ejecución de nuestro programa, todo esto se ve en la figura 15.

The figure shows a terminal window for the server node (0). The output displays the checksum of the matrix C and the command to run the program on the client nodes:

```

975 615 255 -105 -465 -825 -1185 -1545 -1905 -2265

Checksum: -44250
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ git pull
Username for 'https://github.com': EdmundoSanchezM
Password for 'https://EdmundoSanchezM@github.com':
Already up to date.
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ nano MatricesDistribuida.java
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ git pull
Username for 'https://github.com': EdmundoSanchezM
Password for 'https://EdmundoSanchezM@github.com':
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 1), reused 4 (delta 1), pack-reused 0
Unpacking objects: 100% (4/4), 374 bytes | 374.00 KiB/s, done.
From https://github.com/EdmundoSanchezM/Distribuidos
   9a88c26..567b980  main       -> origin/main
Updating 9a88c26..567b980
Fast-forward
   Tarea 3/MatricesDistribuida.java | 2 +-
   1 file changed, 1 insertion(+), 1 deletion(-)
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ nano MatricesDistribuida.java
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ javac MatricesDistribuida.java
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$ java MatricesDistribuida 0 20.85.213.130

Checksum: -4422096843750000
edmundojsm@M2019630428-0:~/Distribuidos/Tarea 3$

```

Figura 15: Final de la ejecución del programa en el nodo 0. Parte 1.

Como vemos solo se nos despliega el valor del checksum de la matriz C, que es como se nos pide en la tarea.

## 2.3. Código

A continuación se anexa el código creado para el cumplimiento de esta tarea, notar que N es estático por lo que para probar diferentes casos de N necesitamos cambiar el valor de N en el programa y volver a compilar.

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.net.ServerSocket;
import java.net.Socket;

class MatricesDistribuida {

    static Object obj = new Object();
    static int N = 10;
    static long[][] A = new long[N][N];
    static long[][] B = new long[N][N];
    static long[][] C = new long[N][N];

    static class Worker extends Thread {

        Socket conexion;

        Worker(Socket conexion) {
            this.conexion = conexion;
        }

        public void run() {
            try {
                DataInputStream entrada = new DataInputStream(conexion.
                    getInputStream());
                DataOutputStream salida = new DataOutputStream(conexion.
                    getOutputStream());
                //Recibimos el numero del nodo que se ejecuto desde el cliente
                int x = entrada.readInt();
                if (x == 1) {
                    for (int i = 0; i < N / 2; i++) {
                        for (int j = 0; j < N; j++) {
                            salida.writeLong(A[i][j]); //Enviar la matriz A1 al
                                nodo 1
                        }
                    }
                    for (int i = 0; i < N / 2; i++) {
                        for (int j = 0; j < N; j++) {
                            salida.writeLong(B[i][j]); //Enviar la matriz B1 al
                                nodo 1
                        }
                    }
                } else if (x == 2) {
                    for (int i = 0; i < N / 2; i++) {
                        for (int j = 0; j < N; j++) {
```

```

        salida.writeLong(A[i][j]); //Enviar la matriz A1 al
            nodo 2
    }
}
for (int i = N / 2; i < N; i++) {
    for (int j = 0; j < N; j++) {
        salida.writeLong(B[i][j]); //Enviar la matriz B2 al
            nodo 2
    }
}
} else if (x == 3) {
    for (int i = N / 2; i < N; i++) {
        for (int j = 0; j < N; j++) {
            salida.writeLong(A[i][j]); //Enviar la matriz A2 al
                nodo 3
        }
    }
    for (int i = 0; i < N / 2; i++) {
        for (int j = 0; j < N; j++) {
            salida.writeLong(B[i][j]); //Enviar la matriz B1 al
                nodo 3
        }
    }
} else if (x == 4) {
    for (int i = N / 2; i < N; i++) {
        for (int j = 0; j < N; j++) {
            salida.writeLong(A[i][j]); //Enviar la matriz A2 al
                nodo 4
        }
    }
    for (int i = N / 2; i < N; i++) {
        for (int j = 0; j < N; j++) {
            salida.writeLong(B[i][j]); //Enviar la matriz B2 al
                nodo 4
        }
    }
}

synchronized (obj) {
    if (x == 1) { //Nodo 1
        //Recibe la matriz C1 del nodo 1
        for (int i = 0; i < (N / 2); i++) {
            for (int j = 0; j < (N / 2); j++) {
                C[i][j] = entrada.readLong();
            }
        }
    } else if (x == 2) { //Nodo 2
        //Recibe la matriz C2 del nodo 2
        for (int i = 0; i < (N / 2); i++) {
            for (int j = (N / 2); j < N; j++) {
                C[i][j] = entrada.readLong();
            }
        }
    } else if (x == 3) { //Nodo 3
        //Recibe la matriz C3 del nodo 3
        for (int i = (N / 2); i < N; i++) {

```

```

        for (int j = 0; j < (N / 2); j++) {
            C[i][j] = entrada.readLong();
        }
    }
} else if (x == 4) { //Nodo 4
    //Recibe la matriz C4 del nodo 4
    for (int i = (N / 2); i < N; i++) {
        for (int j = (N / 2); j < N; j++) {
            C[i][j] = entrada.readLong();
        }
    }
}
}
entrada.close();
salida.close();
conexion.close();
} 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 en el anillo");
        System.exit(1);
    }

    int nodo = Integer.valueOf(args[0]);
    String ip = args[1];
    if (nodo == 0) {
        //Iniciando el servidor
        ServerSocket servidor = new ServerSocket(20000);
        //Iniciando A y B
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++) {
                A[i][j] = i + 2 * j;
                B[i][j] = i - 2 * j;
                C[i][j] = 0;
            }
        }
        if (N == 10) {
            System.out.println("\nValor Matriz A\n");
            ImprimirMatriz(A);
            System.out.println("\nValor Matriz B\n");
            ImprimirMatriz(B);
        }
        //Transpone la matriz B, la matriz traspuesta queda en B
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < i; j++) {
                long x = B[i][j];
                B[i][j] = B[j][i];
                B[j][i] = x;
            }
        }
    }
}

```



```

Worker v[] = new Worker[4]; //Aceptaremos 4 clientes
int nodosClientes = 0;
//Espera y aceptacion de cada nodo cliente que se va conectar al
    nodo servidor
while (nodosClientes != 4) {
    Socket conexion = servidor.accept();
    v[nodosClientes] = new Worker(conexion);
    v[nodosClientes].start();
    nodosClientes++;
}
nodosClientes = 0;
//Esperamos a que los nodos terminen para avanzar.
while (nodosClientes < 4) {
    v[nodosClientes].join();
    nodosClientes++;
}
//Inicializamos la variable checksum
long checksum = 0;
for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
        checksum += C[i][j];
    }
}
if (N == 10) {
    System.out.println("\nValor Matriz C\n");
    ImprimirMatriz(C);
}
System.out.println("\nChecksum: " + checksum);
} else { // Nodos clientes
    //Creamos la conexion
    Socket conexion = null;
    for (;;)
        try {
            conexion = new Socket(ip, 20000);
            break;
        } catch (Exception e) {
            Thread.sleep(100);
        }
    DataInputStream entrada = new DataInputStream(conexion.
        getInputStream());
    DataOutputStream salida = new DataOutputStream(conexion.
        getOutputStream());
    //Enviamos el numero del nodo en que se ejecuta el cliente al
        servidor
    salida.writeInt(nodo);
    //Declaramos Matriz A2 y Matriz B2, guardaran las mitades recibidas
    long[][] A2 = new long[N / 2][N];
    long[][] B2 = new long[N / 2][N];
    //Declaramos Matriz Cc, guardaran los cuartos de C
    long[][] Cc = new long[N / 2][N / 2];
    //Reciben An y Bn, donde n puede ser 1 o 2
    for (int i = 0; i < (N / 2); i++) {
        for (int j = 0; j < N; j++) {
            A2[i][j] = entrada.readLong(); //Recibimos la mitad de A.
        }
    }
}

```

```

        for (int i = 0; i < (N / 2); i++) {
            for (int j = 0; j < N; j++) {
                B2[i][j] = entrada.readLong();//Recibimos la mitad de B.
            }
        }
        //Cn=AmitadxBmitad (renglon por renglon)
        for (int i = 0; i < (N / 2); i++) {
            for (int j = 0; j < (N / 2); j++) {
                for (int k = 0; k < N; k++) {
                    Cc[i][j] += A2[i][k] * B2[j][k];
                }
            }
        }
        //Enviar la matriz Cn al nodo 0
        for (int i = 0; i < (N / 2); i++) {
            for (int j = 0; j < (N / 2); j++) {
                salida.writeLong(Cc[i][j]);
            }
        }
        entrada.close();
        salida.close();
        conexion.close();
    }
}

public static void ImprimirMatriz(long[][] matriz) {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            System.out.print(matriz[i][j] + "\t");
        }
        System.out.println();
    }
}
}

```

### 3. Conclusiones

El desarrollo del código de la práctica fue muy interesante ya que es la primera vez en toda mi instancia en ESCOM en la que uso una maquina virtual remota, en este caso de Azure y pude notar que es tener practica una maquina con Ubuntu pero sin interfaz gráfica y solo podemos usar los comandos que tiene Linux, ahora viendo la parte de la implementación de la practica no fue muy complicada ya que teníamos experiencia anterior con este tipo de topologia y requisitos, lo mas interesante sin duda de esta practica en mi opinión fueron dos cosas la primera el uso de maquinas virtuales remotas y la segunda fue el como enviar las partes de las matriz a los nodos, en mi caso mande numero a numero el contenido de las matrices, otra forma y la cual podría ser la mas simple seria mandar toda la parte de la matriz que se solicita como un objeto, sin embargo, considere que lo mejor seria mandar elemento a elemento del arreglo ya que considero que es mas eficiente.