Instituto Politécnico Nacional.
Escuela Superior de Cómputo
(ESCOM).

Desarrollo de Sistemas Distribuidos.

Tarea 6.
“Multiplicación de matrices utilizando objetos distribuidos”.

Profesor Carlos Pineda
Alumno: Sánchez Martínez Eli
Grupo: 4CM5

20/11/2020

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

Desarrolló:

Al igual que la tarea 3, se hará la multiplicación de matrices distribuidas, pero esta vez se ejecutará usando RMI, primeramente, se deberá implementar la siguiente topología de red.

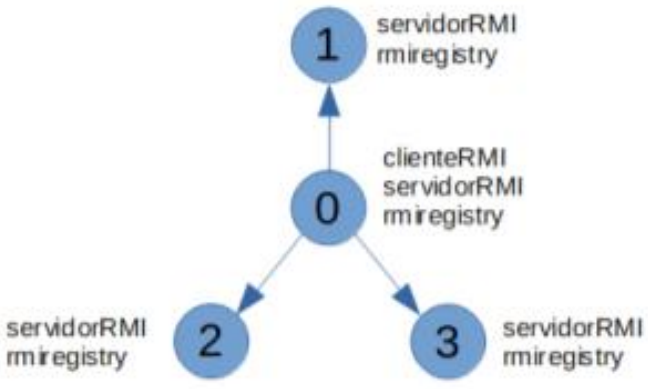
Se deberá ejecutar dos casos:

- N=4, se deberá desplegar las matrices A, B y C y el checksum de la matriz C.
- N=500, deberá desplegar el checksum de la matriz C.

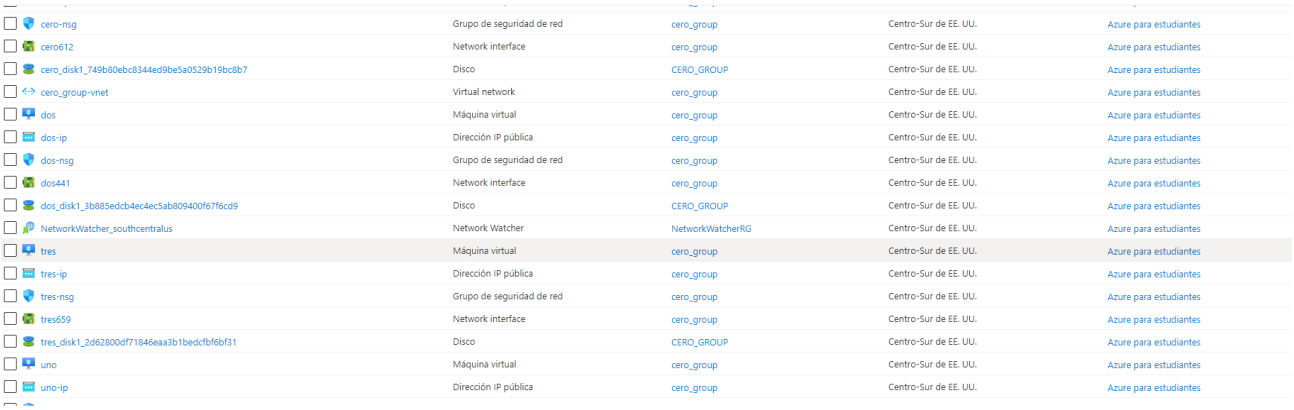
Los elementos de las matrices serán de tipo int y el checksum será de tipo long.

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]=2 * i - j
- B[i][j] = 2 * i + j



Para poder realizar la práctica es importante destacar que las máquinas virtuales deben crearse en el mismo grupo de recursos, a modo que se pueda utilizar su dirección IP privada para comunicarse entre ellas.



cero-nsg	Grupo de seguridad de red	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
cero612	Network interface	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
cero_disk1_749b80ebc8344ed9be5a0529b19bc8b7	Disco	CERO_GROUP	Centro-Sur de EE. UU.	Azure para estudiantes
cero_group-vnet	Virtual network	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
dos	Máquina virtual	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
dos-ip	Dirección IP pública	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
dos-nsg	Grupo de seguridad de red	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
dos441	Network interface	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
dos_disk1_3b885edcb4ec4ec5ab809400f67f6cd9	Disco	CERO_GROUP	Centro-Sur de EE. UU.	Azure para estudiantes
NetworkWatcher_southcentralus	Network Watcher	NetworkWatcherRG	Centro-Sur de EE. UU.	Azure para estudiantes
tres	Máquina virtual	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
tres-ip	Dirección IP pública	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
tres-nsg	Grupo de seguridad de red	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
tres659	Network interface	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
tres_disk1_2d62800df71846eaa3b1bedcfbf6bf31	Disco	CERO_GROUP	Centro-Sur de EE. UU.	Azure para estudiantes
uno	Máquina virtual	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes
uno-ip	Dirección IP pública	cero_group	Centro-Sur de EE. UU.	Azure para estudiantes

1.*Recursos creados para cada VM.*

Una vez que tenemos las máquinas listas para usarse, se procede a explicarse el proceso que se hizo en cada máquina virtual con una máquina de ejemplo.

```
C:\Users\elit>ssh dos@13.85.85.7
The authenticity of host '13.85.85.7 (13.85.85.7)' can't be established.
ECDSA key fingerprint is SHA256:DDh5Phglmdw7vHsJzXt519Xs886JvBJ9oY7wPWAvdhM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.85.85.7' (ECDSA) to the list of known hosts.
dos@13.85.85.7's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1031-azure x86_64)
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1031-azure x86_64)

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

  System information as of Sat Nov 21 04:03:34 UTC 2020

  System load:  0.39              Processes:           113
  Usage of /:   4.4% of 28.90GB   Users logged in:     0
  Memory usage: 20%               IP address for eth0: 10.0.0.6
  Swap usage:   0%

0 packages can be updated.
0 updates are security updates.
```

2. *Ingresando a cada máquina virtual.*

Procedemos a cargar los archivos a cada máquina virtual:

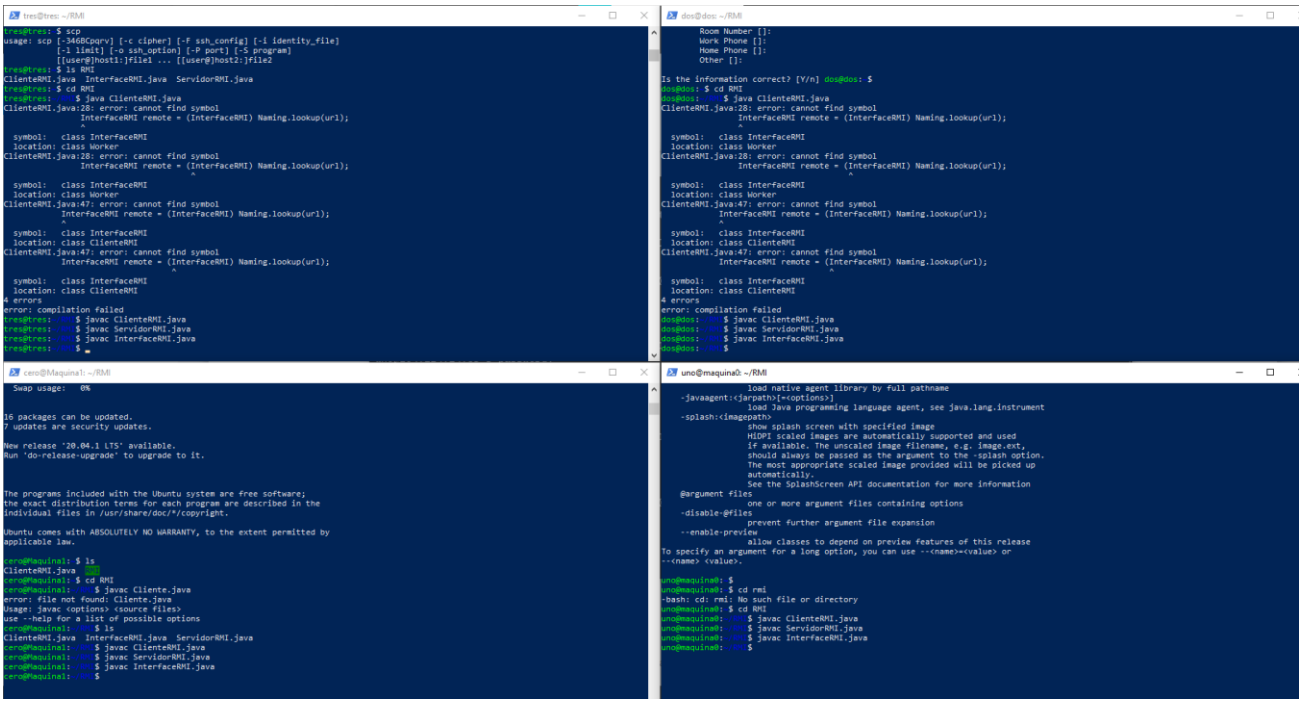
```
C:\Users\elit\Desktop\ESCOM_ELI\Trabajos 6>scp -r .\RMI cero@104.215.75.1:~
cero@104.215.75.1's password:
Permission denied, please try again.
cero@104.215.75.1's password:
cinco.png                                     100% 7686    11.3KB/s   00:00
cuetro.png                                    100%  239KB  59.7KB/s   00:04
dos.png                                       100%  182KB 181.3KB/s   00:01
C:\Users\elit\Desktop\ESCOM_ELI\Trabajos 6>scp -r .\RMI cero@104.215.75.1:~
cero@104.215.75.1's password:
ClientRMI.java                                100% 5566    39.6KB/s   00:00
InterfaceRMI.java                             100%  189     3.2KB/s   00:00
Makefile                                      100%  610     9.3KB/s   00:00
ServerRMI.java                                100% 1947    22.7KB/s   00:00

C:\Users\elit\Desktop\ESCOM_ELI\Trabajos 6>scp -r .\RMI uno@104.214.73.25:~
uno@104.214.73.25's password:
ClientRMI.java                                100% 5566    39.5KB/s   00:00
InterfaceRMI.java                             100%  189     3.2KB/s   00:00
Makefile                                      100%  610     9.2KB/s   00:00
ServerRMI.java                                100% 1947    23.1KB/s   00:00

C:\Users\elit\Desktop\ESCOM_ELI\Trabajos 6>scp -r .\RMI dos@13.85.85.7:~
dos@13.85.85.7's password:
ClientRMI.java                                100% 5566     8.0KB/s   00:00
InterfaceRMI.java                             100%  189     3.1KB/s   00:00
Makefile                                      100%  610     9.2KB/s   00:00
ServerRMI.java                                100% 1947    21.8KB/s   00:00

C:\Users\elit\Desktop\ESCOM_ELI\Trabajos 6>scp -r .\RMI tres@104.214.59.92:~
tres@104.214.59.92's password:
ClientRMI.java                                100% 5566    10.1KB/s   00:00
InterfaceRMI.java                             100%  189     3.1KB/s   00:00
Makefile                                      100%  610     9.2KB/s   00:00
ServerRMI.java                                100% 1947    22.4KB/s   00:00
```

***3**. Cargando los archivos a cada máquina virtual.*

Procedemos a compilar los archivos en cada VM y ejecutamos los comandos, primero compilamos los archivos con terminación java, con el comando:

- javac nombre_archivo.java
- java nombre_archivo.

Una vez compilados procedemos a ejecutar cada archivo con su respectivo nodo para poder ver la implementacion de nuestra multiplicación.



***4**.Compilando todos los archivos con terminación java en las VM.*

Para ejecutar tenemos que poner en cada nodo nuestros ejecutables de java con terminación:

- InterfaceRMI.java
- ServerRMI.java

Y para finalizar ejecutamos nuestro Server con el numero de N y el puerto en este caso, este proceso se repite en 3 nodos.

```
cero@cero:~/RMI$ javac InterfaceRMI.java
cero@cero:~/RMI$ javac ServerRMI.java
cero@cero:~/RMI$ java ServerRMI 4 1099
rmi://localhost:1099/prueba
```

***5**. Compilando los archivos en un nodo de la VM.*

En el cuarto nodo ejecutamos el ClienteRMI.java, dándole nuestro valor de N y también la dirección IP junto con su respectivo puerto que en este caso es de 1099 y observamos el resultado de la multiplicación de nuestra matriz.

```
tres@tres:~/RMI$ javac ClientRMI.java
tres@tres:~/RMI$ java ClientRMI 4 104.215.75.1:1099 104.214.73.25:1099 13.85.85.7:1099 104.214.59.92:1099
A:
0 -1 -2 -3
2 1 0 -1
4 3 2 1
6 5 4 3
B:
0 1 2 3
2 3 4 5
4 5 6 7
6 7 8 9
El checksum es: 272
C:
-28 -34 -40 -46
-4 -2 0 2
20 30 40 50
44 62 80 98
```

6. *Resultado del cliente de una VM.*

Ejecutamos el siguiente caso, en el cual N vale 500, entonces se repite el mismo proceso que con el valor de N 4, solo que se cambia por 500.

```
^Ccero@cero:~/RMI$ java ServerRMI 500 1099
rmi://localhost:1099/prueba
```

7. *Ejecutando el serverRMI en una VM.*

Al hacer eso, solo se tendrá que imprimir el checksum, el cual podemos ver a continuación:

```
tres@tres:~/RMI$ java ClientRMI 500 104.215.75.1:1099 104.214.73.25:1099 13.85.85.7:1099 104.214.59.92:1099
El checksum es: 18135531250000
```

8*.Resultado del cliente en la VM.*

Conclusión.

En esta práctica se puede notar a la perfección la diferencia entre usar objetos remotos, ya que esta misma actividad se realizó en la tarea 3, usar sockets. Esta hizo que fuera una tarea más complicada ya que él envió de los datos lo hacían una tarea complicada.

El usar objetos remotos nos garantiza una forma más sencilla de realizar la tarea además de que la integridad de los datos se garantiza de una mejor manera en contraste a enviar datos por sockets, incluso esto se ve reflejado en la cantidad de líneas de código que se usaron para el programa.