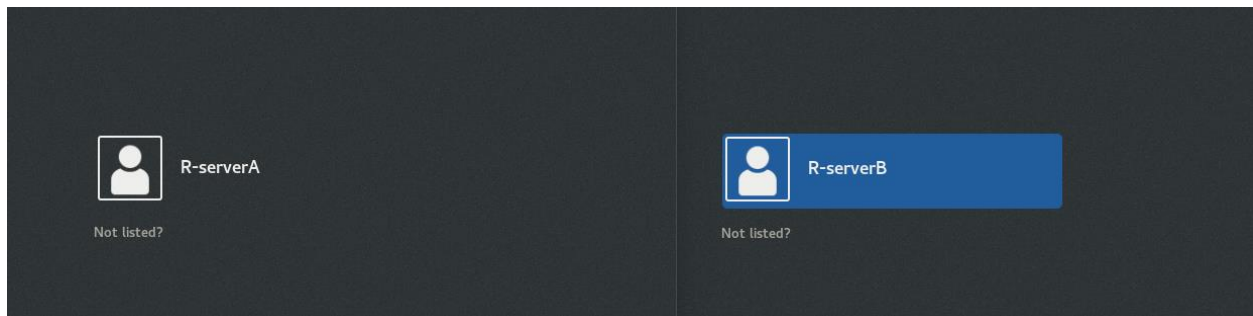


Creation of centos server VMs is done:



Making the two machines communicate:

◆ R-serverA IP address:

```
[r-serverA@localhost ~]$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.217 netmask 255.255.255.0 broadcast 192.168.100.255
    inet6 fe80::c894:6b8b:7912:de5 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:87:0c:fc txqueuelen 1000 (Ethernet)
    RX packets 25808 bytes 34735906 (33.1 MiB)
    RX errors 0 dropped 19 overruns 0 frame 0
    TX packets 1918 bytes 131612 (128.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

◆ R-serverB IP address:

```
[r-serverB@localhost ~]$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.218 netmask 255.255.255.0 broadcast 192.168.100.255
    inet6 2a02:2908:5102:98f3:8ba:4c2a:181a:6b07 prefixlen 64 scopeid 0x0<global>
    inet6 fe80::e12b:bad1:7071:be94 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:04:1f:73 txqueuelen 1000 (Ethernet)
    RX packets 510928 bytes 749530717 (714.8 MiB)
    RX errors 0 dropped 20 overruns 0 frame 0
    TX packets 52321 bytes 3252096 (3.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

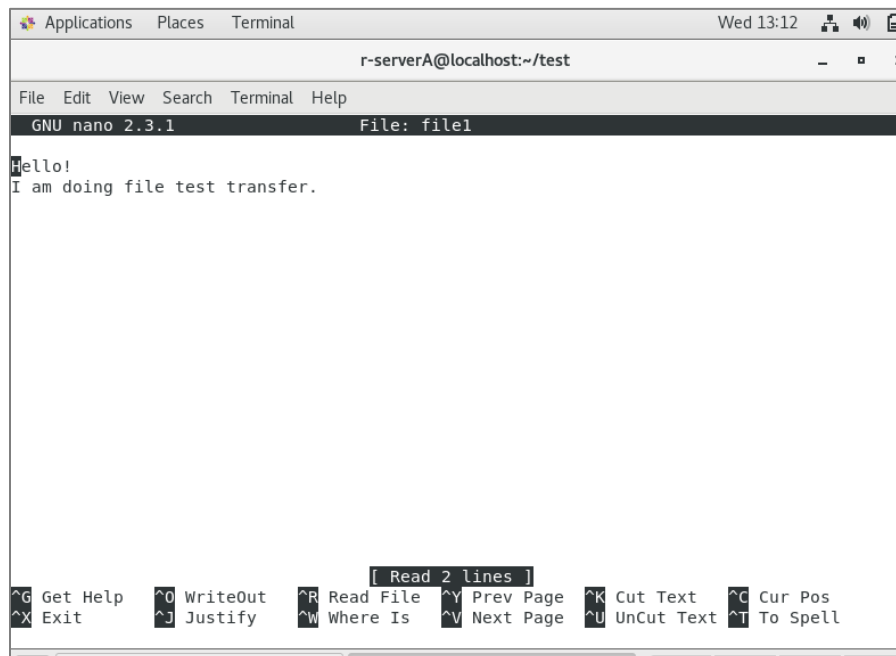
◆ **Server A communicating to Server B using ping command:**

```
[r-serverA@localhost ~]$ ping 192.168.100.218
PING 192.168.100.218 (192.168.100.218) 56(84) bytes of data.
64 bytes from 192.168.100.218: icmp_seq=1 ttl=64 time=4.92 ms
64 bytes from 192.168.100.218: icmp_seq=2 ttl=64 time=1.19 ms
64 bytes from 192.168.100.218: icmp_seq=3 ttl=64 time=2.16 ms
64 bytes from 192.168.100.218: icmp_seq=4 ttl=64 time=2.08 ms
64 bytes from 192.168.100.218: icmp_seq=5 ttl=64 time=1.62 ms
^Z
[3]+  Stopped                  ping 192.168.100.218
[r-serverA@localhost ~]$ █
```

◆ **Creating Directory:**

```
[r-serverA@localhost ~]$ mkdir test
[r-serverA@localhost ~]$ cd test
[r-serverA@localhost test]$ nano file1
[r-serverA@localhost test]$ █
```

◆ **Creating File:**



```
Applications  Places  Terminal  Wed 13:12
r-serverA@localhost:~/test
File Edit View Search Terminal Help
GNU nano 2.3.1 File: file1
Hello!
I am doing file test transfer.

[ Read 2 lines ]
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

◆ Changing the permissions for “test” directory:

```
[r-serverA@localhost ~]$ chmod 704 test
[r-serverA@localhost ~]$ ls -l
total 0
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Desktop
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Documents
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Downloads
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Music
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Pictures
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Public
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Templates
drwx---r--. 2 r-serverA r-serverA 19 Dec  6 13:03 test
drwxr-xr-x. 2 r-serverA r-serverA  6 Dec  6 12:06 Videos
[r-serverA@localhost ~]$
```

◆ Compressing the directory using tar command:

-c → means create archive.

-z → means the type of compression is gzip.

-v → means verbose, which basically means output the process of the command.

-f → means filename.

Then we specify both the new archive file name and then the name of directory or file we want to compress.

```
[r-serverA@localhost ~]$ tar -czvf transfercom.tar.gz test
test/
test/file1
[r-serverA@localhost ~]$ ls
Desktop    Downloads  Pictures   Templates  transfercom.tar.gz
Documents  Music      Public     test        Videos
[r-serverA@localhost ~]$
```

SCP protocol

This protocol is used to transfer files securely from one device to another. By default the scp protocol transfers the file encrypted. But you can specify what type of encryption using the -c option, which means cipher, and select the algorithm you want to encrypt the transfer of files.

In the below figure, I'll break down the command into smaller sectors as I applied more than one concept.

```
[r-serverA@localhost ~]$ (scp -P 22 -v transfercom.tar.gz r-serverB@:/home/r-serverB ; date ; tar -tf transfercom.tar.gz) > logactivity.txt
Executing: program /usr/bin/ssh host _____, user r-serverB, command scp -v -t /home/r-serverB
OpenSSH_7.4p1, OpenSSL 1.0.2k-fips 26 Jan 2017
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 58: Applying options for *
debug1: Connecting to _____ port 22.
debug1: Connection established.
```

◆ First scp command:

scp -P 22 -v transfercom.tar.gz username@ip-address:/home/user-directory	
-P 22	-P means to specify port. I wrote 22 which refers to SSH tcp port.
-v	Displaying the process of the command.
Transfercom.tar.gz	Filename to be transferred.
username@ip-address	Username @ that machines address.
/home/userdirectory	Transfer to this directory.

To ensure that the transfer is encrypted, I purposely typed the option -v, to display how the command process goes. As you can see in the below screenshot, the SSH-2.0 is enabled. Also you can see the process of establishing the session between the two servers.

```

debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_7.4
debug1: Remote protocol version 2.0, remote software version OpenSSH_7.4
debug1: match: OpenSSH_7.4 pat OpenSSH* compat 0x04000000
debug1: Authenticating to          :22 as 'r-serverB'
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: curve25519-sha256
debug1: kex: host key algorithm: ecdsa-sha2-nistp256
debug1: kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compr
ession: none
debug1: kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compr
ession: none
debug1: kex: curve25519-sha256 need=64 dh_need=64
debug1: kex: curve25519-sha256 need=64 dh_need=64
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: ecdsa-sha2-nistp256 SHA256:oj3VKTY+5NMv/+3LS8JQCzCi0x1jws9AZjY
ND5qDi68
debug1: Host '          ' is known and matches the ECDSA host key.
debug1: Found key in /home/r-serverA/.ssh/known_hosts:1
debug1: rekey after 134217728 blocks
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: SSH2_MSG_NEWKEYS received
debug1: rekey after 134217728 blocks
debug1: SSH2_MSG_EXT_INFO received

```

Once the transfer is successful it is requested from us to implement a log file for the activity, along with displaying the time and date, and the files that are compressed. And save the output from this command in the file created. **Note: the scp output won't be logged. Scp is known that it does not create any logs when transferring the files.**

```

Transferred: sent 2356, received 2484 bytes, in 0.5 seconds
Bytes per second: sent 4916.0, received 5183.0
debug1: Exit status 0
[r-serverA@localhost ~]$ cat logactivity.txt
Wed Dec  6 14:31:01 +04 2023
test/
test/file1
[r-serverA@localhost ~]$ █

```

The other half of the command I wrote is:

; date ; tar -tf filename.tar.gz) > logactivity.txt	
date	Displays the day, date and time.
Tar -tf filename.tar.gz	To display what is inside the archive.
()	The parenthesis is used to ensure that all the commands output is then saved in the file, which is directed using >
> logactivity.txt	Directing the output to be saved in file.

So the full form the issued command is:

```
(scp -P 22 -v transfercom.tar.gz username@ip-address:/home/user-directory ; date ; tar -tf transfercom.tar.gz) > logactivity.txt
```

◆ **Checking if the transfer is successful in server B:**

```
[r-serverB@localhost ~]$ ls -l
total 4
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Desktop
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Documents
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Downloads
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Music
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Pictures
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Public
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Templates
-rw-rw-r--. 1 r-serverB r-serverB 187 Dec  6 13:15 transfercom.tar.gz
drwxr-xr-x. 2 r-serverB r-serverB  6 Dec  6 10:28 Videos
[r-serverB@localhost ~]$ tar -xzf transfercom.tar.gz
test/
test/file1
```

◆ **Changing the permissions of the file and keep what is needed:**

```
[r-serverB@localhost ~]$ cd test
[r-serverB@localhost test]$ ls -l
total 4
-rw-rw-r--. 1 r-serverB r-serverB 39 Dec  6 13:03 file1
[r-serverB@localhost test]$ chmod 400 file1
[r-serverB@localhost test]$ ls -l
total 4
-r-----. 1 r-serverB r-serverB 39 Dec  6 13:03 file1
[r-serverB@localhost test]$ █
```