Sprint 3 ASIST

Table of contents

[Context 2](#_Toc123920748)

[SSH Access by certificate only and for the administrator only 2](#_Toc123920749)

[Setting up the connection by certificate 2](#_Toc123920750)

[Generation of public and private keys for the administrator 2](#_Toc123920751)

[Configuration of the SSH connection through these keys 3](#_Toc123920752)

[Modification of the rights on the private key 4](#_Toc123920753)

[Restart of the SSH Service 4](#_Toc123920754)

[Disable password login 4](#_Toc123920755)

[Verification of correct operation 4](#_Toc123920756)

[Conclusion 5](#_Toc123920757)

[Setting up a SMB public file share 6](#_Toc123920758)

[Installation and configuration of Samba. 6](#_Toc123920759)

[Configuration of the public share 7](#_Toc123920760)

[Creation of a user "student1220379" and the group "isepTeams 8](#_Toc123920761)

[Accessing the Share 10](#_Toc123920762)

[Conclusion 12](#_Toc123920763)

# Context

We chose the following user cases :

* Number 10: « As a system administrator I want the administrator to have SSH access to the virtual machine, by certificate only, without using a password.»
* Number 11: « As a system administrator I want a public file share, SMB/CIFS or NFS format, to be created to speed up the process among the various teams.»

The operations will be operated on a machine running Debian 11.

# SSH Access by certificate only and for the administrator only

## Setting up the connection by certificate

In order not to be blocked without any means of access to the SSH service of our server, we need to configure an alternative authentication method before disabling the password authentication. We will therefore configure authentication by certificate, or key pair.

### Generation of public and private keys for the administrator

In order to generate a key pair to authenticate the administrator to the SSH service, we must first open a shell with his account.

Once we have a root shell, we can navigate to the /root/.ssh directory with $ cd /root/.ssh (or create it with $ mkdir /root/.ssh).

Then we create the key pair with the command $ ssh-keygen -b 4096. The -b option is used to specify the length of the key.

We press the Enter key to validate the default location "/root/.ssh/id\_rsa", then as we don't want passwords associated with the keys, we press Enter twice.

Une image contenant texte

Description générée automatiquement

Figure - Creation of the private and public keys

Two files are then created:

* id\_rsa: This is the private key, to be kept secret and that only the administrator must hold
* id\_rsa.pub: This is the public key associated with the private key, It is present on the server in order to authorize SSH access by the private key.

Une image contenant texte, orange, sombre, image

Description générée automatiquement

Figure - Contents of the /root/.ssh folder

### Configuration of the SSH connection through these keys

Once the key pair is created, we need to authorize it in order to open an SSH session with the root user.

To do this, we need to write the contents of the public key to the authorized\_keys file with the command cat id\_rsa.pub > authorized\_keys.

Une image contenant texte

Description générée automatiquement

Figure - Contents of the authorized\_keys file after copying the contents of the public key into it

### Modification of the rights on the private key

In order to use the private key when authenticating to the SSH session, we need to reduce its rights. To do this, we use the command chmod 600 id\_rsa. This means that only the owner of the file can read and write to it, but he cannot execute it, and other users cannot do anything with it.

### Restart of the SSH Service

In order to make all our modifications operational, we just have to restart the SSH service with the command systemctl restart sshd.

## Disable password login

The connection by certificate being now functional, we only have to disable the connection by password.

To do this, we go to the file and replace **#PassworAuthentication yes** with **PasswordAuthentication no**.

### Verification of correct operation

#### Test of the connection by certificate to the root account

Une image contenant texte

Description générée automatiquement

Figure - Creation a root SSH Session with certificate

Our previously created certificate is accepted when creating an SSH session to our server's root account from our local machine.

#### Testing the connection to the root account with password

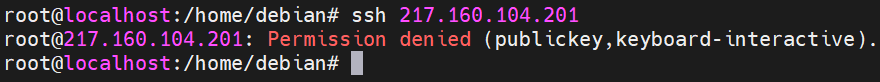


Figure - Our server refuses our attempt to connect to the root account by password

Our server denies SSH login to the root account when we log in with a password and not with a certificate.

### Conclusion

Having disabled password login and enabled certificate login for the root account, the router now denies SSH logins from users other than root, and denies SSH logins on the root account by password.

# Setting up a SMB public file share

So that each member of the team can exchange files easily and quickly, we will set up an SMB file sharing system with Samba.

## Installation and configuration of Samba.

First, we begin to install Samba. We can do this with the command:

apt-get install samba

Une image contenant texte

Description générée automatiquement

Figure - Beginning of the installation of Samba

Une image contenant texte

Description générée automatiquement

Figure - End of the installation of Samba

Then, we ask the system to run the service automatically with the command

Systemctl enable smbd

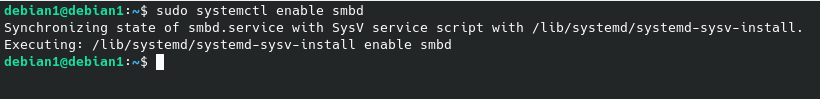


Figure - Execution of the command that enables Samba

Une image contenant texte

Description générée automatiquement

Figure - Status of the Samba service

## Configuration of the public share

Now that Samba is installed and running automatically, we need to configure a share in the configuration file. This file is called "smb.conf" and is located in the "/etc/samba/" directory.

We then edit this file and add the following lines:

[isepShare][[1]](#footnote-1)

comment = Shared Folder

path = /srv/share [[2]](#footnote-2)

guest ok = no [[3]](#footnote-3)

read only = no [[4]](#footnote-4)

browseable = yes

valid users = @isepTeams[[5]](#footnote-5)

Une image contenant texte

Description générée automatiquement

Figure - Content of the smb.conf file

### Creation of a user "student1220379" and the group "isepTeams

To access the samba share, we need to create the group isepteams and add users to it.

Let’s create the user student1220379 with the command

sudo adduser student1220379

Une image contenant texte

Description générée automatiquement

Figure - Creation of the user student1220379

Then we have to create add the user to the samba users with the command

sudo smbpasswd -a student1220379

Une image contenant texte

Description générée automatiquement

Figure - Adding the user student1220379 to the samba users

And we create a group that will contains allowed users to use the share. This with the command sudo groupadd isepTeams

Une image contenant texte

Description générée automatiquement

Figure - Creation of the isepTeams group on the system

Once we have our user and our group, let’s add the user to the group isepTeams with the command

sudo gpasswd -a student1220379 isepTeams

Une image contenant texte

Description générée automatiquement

Figure - Adding the user student1220379 to the group isepTeams

As we will need a folder to represent our share, we will create a new folder dedicated to it called « share » and located in the « /srv » directory :

sudo mkdir /srv/share



Figure - Creation of the share folder in the /srv directory

Now quite everything is ready, It remains to make the group isepTeams owner of the share folder. The -R option make this operation recursive (To all subfolders) :

sudo chgrp -R isepTeams /srv/share



Figure - Definition of the group isepTeams as the owner of the /srv/share folder recursively

Finally, we allow the owner group to read and write inside the share with the command :

sudo chmod -R g+rw /srv/share



Figure - Allowing the group to read and write into the folder

Then, we just have to restart the samba daemon with the command

sudo systemctl restart smbd

## Accessing the Share

Now that our share is set up, we can access It through any machine in the same network. To reach the share with Windows, we just have to acces this address in the navigation bar «\\debian1\isepShare » with debian1 the name of our server in the network and isepShare the name of the share.

Une image contenant texte

Description générée automatiquement

Figure - We can found our share from our local Windows machine

When we try to access the share, a windows prompt us for credentials, we must fill our previously created user and password with (student1220379)

Une image contenant texte

Description générée automatiquement

Figure - We log in with the account previously created

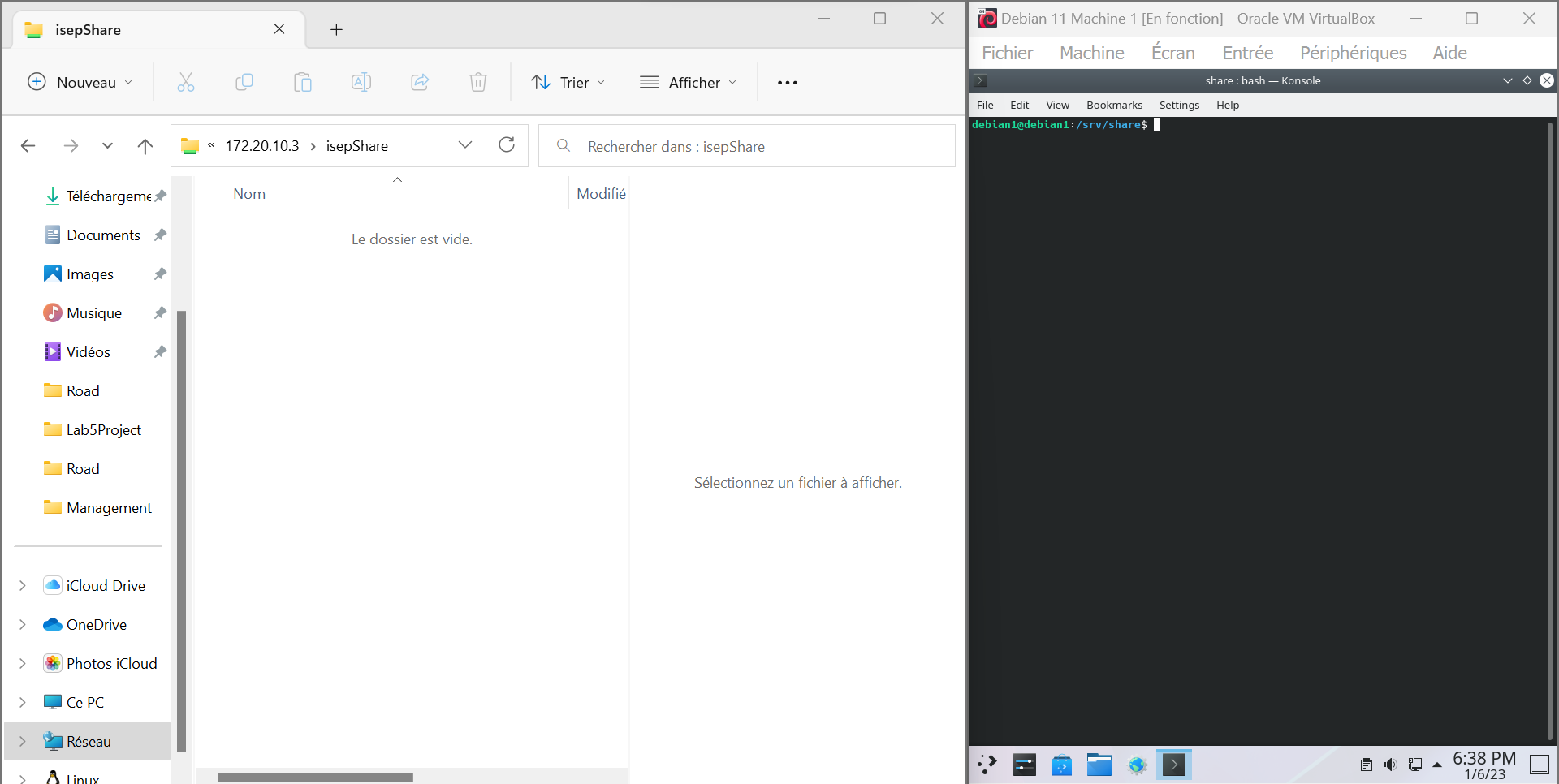


Figure - We manage to enter the Share

We are now in the Share. We can read, add, remove or delete files from here and the modifications will be applied on the share.

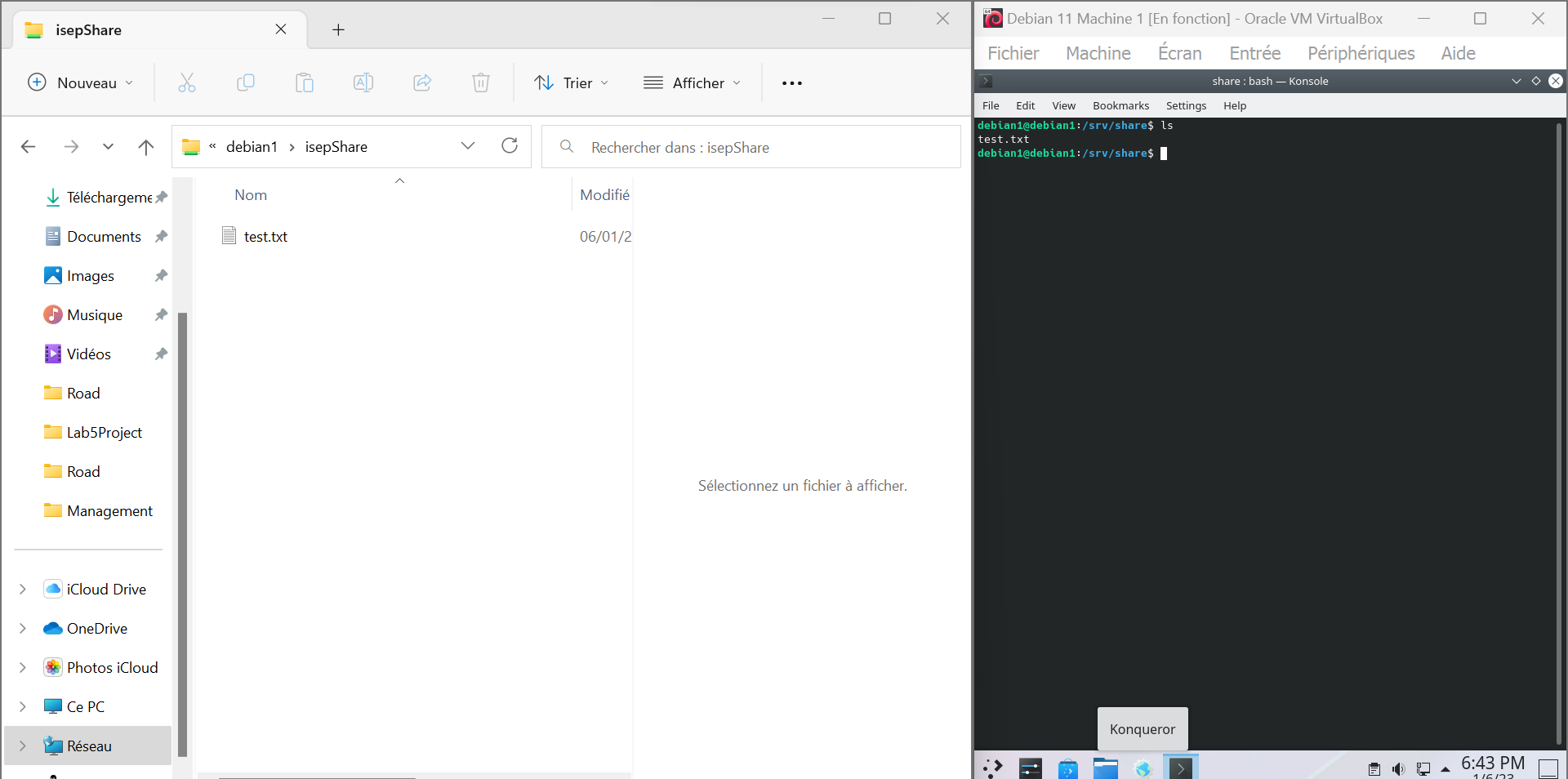


Figure - We can write on the share

Here we created a new file called “test.txt” and when we list he files inside the share directory in the server, we can find this “test.txt” file.

## Conclusion

Now that our share is created and well configured, every member of the group “isepTeams” on the server will be able to access it through the same network. We could access It from anywhere if we opened the port 445 with a whitelist (To avoid attacks).

1. This is the name of the share [↑](#footnote-ref-1)
2. This is the path of the shared folder containing all the shared files [↑](#footnote-ref-2)
3. We don’t allow that anybody can access the share [↑](#footnote-ref-3)
4. We want to be able to modify the files in the share [↑](#footnote-ref-4)
5. All the members belonging the the group « isepTeams » can access and edit the files in the share [↑](#footnote-ref-5)