MySQL TLS Lab

Date: August 8 2025
Machine: Ubuntu Server (VM) on Windows Host
Objective: Capture and analyze MySQL traffic over an encrypted TLS connection.

Overview

The goal was to set up a MySQL server on my Ubuntu VM with TLS encryption, connect to it securely using a custom user, and capture the traffic with `tcpdump` to analyze whether the connection protects sensitive data (like credentials and queries). I also wanted to copy the `.pcap` file to my Windows system for analysis in Wireshark.

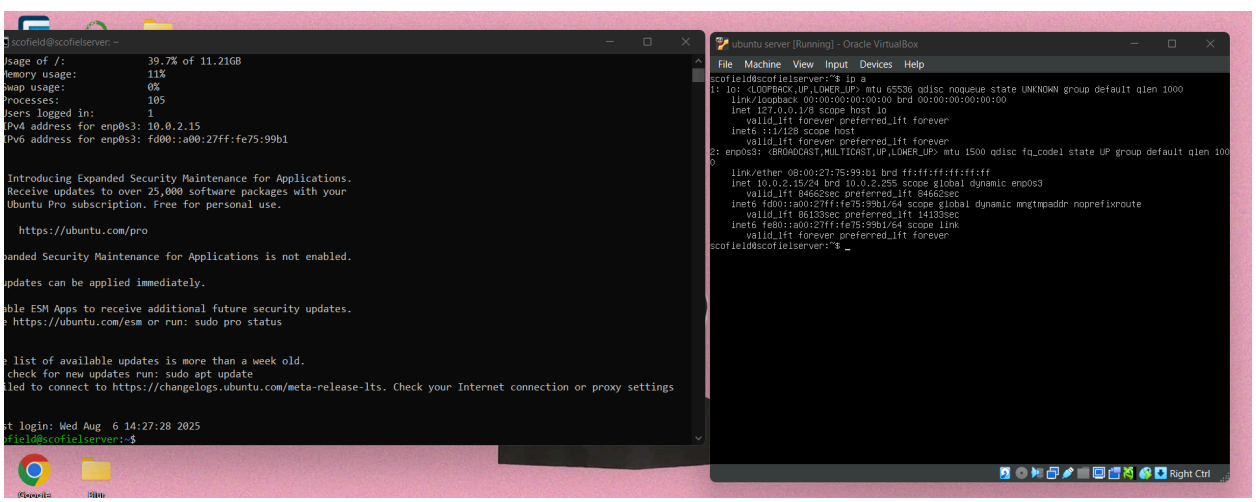
1. SSH Setup

To avoid the copy-paste limitation in the VM terminal, I enabled SSH so I could connect from my Windows CMD or PowerShell.

Commands used:

sudo apt update
sudo apt install openssh-server -y
sudo systemctl enable ssh
sudo systemctl start ssh

Screenshot: SSH server running status



Issue Encountered:

Initially, I was confused about copying and pasting in CMD. I realized I had to enable **QuickEdit Mode** or use **Windows Terminal**, which made the process much easier.

2. MySQL Installation and TLS Setup

I installed MySQL using:

sudo apt install mysql-server -y

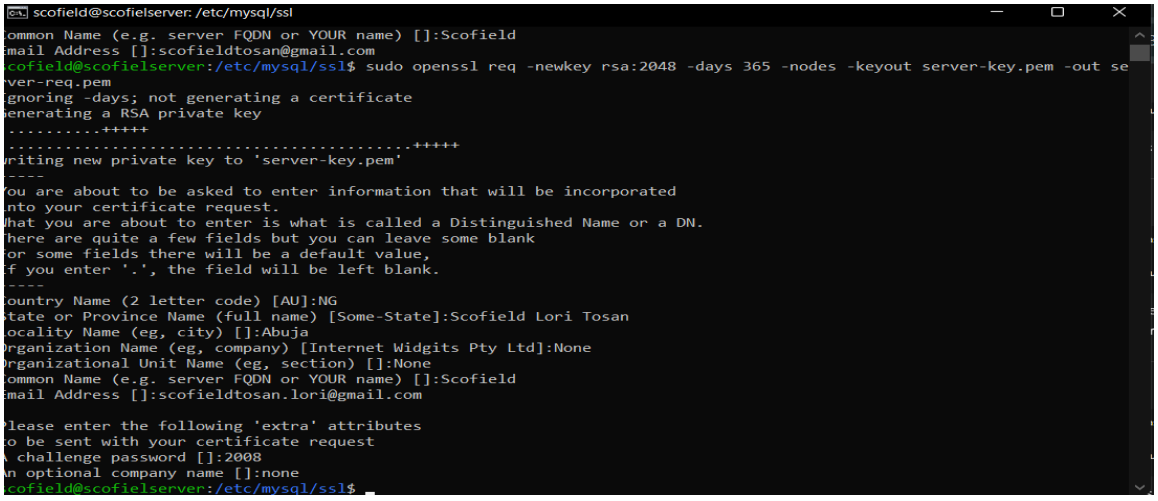
Then I generated custom TLS certificates using `openssl`:

sudo openssl genrsa 2048 | sudo tee ca-key.pem > /dev/null
sudo openssl req -new -x509 -nodes -days 365 -key ca-key.pem -out ca.pem
sudo openssl req -newkey rsa:2048 -days 365 -nodes -keyout server-key.pem -out server-req.pem
sudo openssl x509 -req -in server-req.pem -days 365 -CA ca.pem -CAkey ca-key.pem -set_serial 01 -out server-cert.pem

I moved them to `/etc/mysql/ssl` and fixed permissions:

sudo chmod 600 *.pem
sudo chown mysql:mysql *.pem

Screenshot: Certificate files in `/etc/mysql/ssl`



3. MySQL Configuration

I updated the MySQL config file:

sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

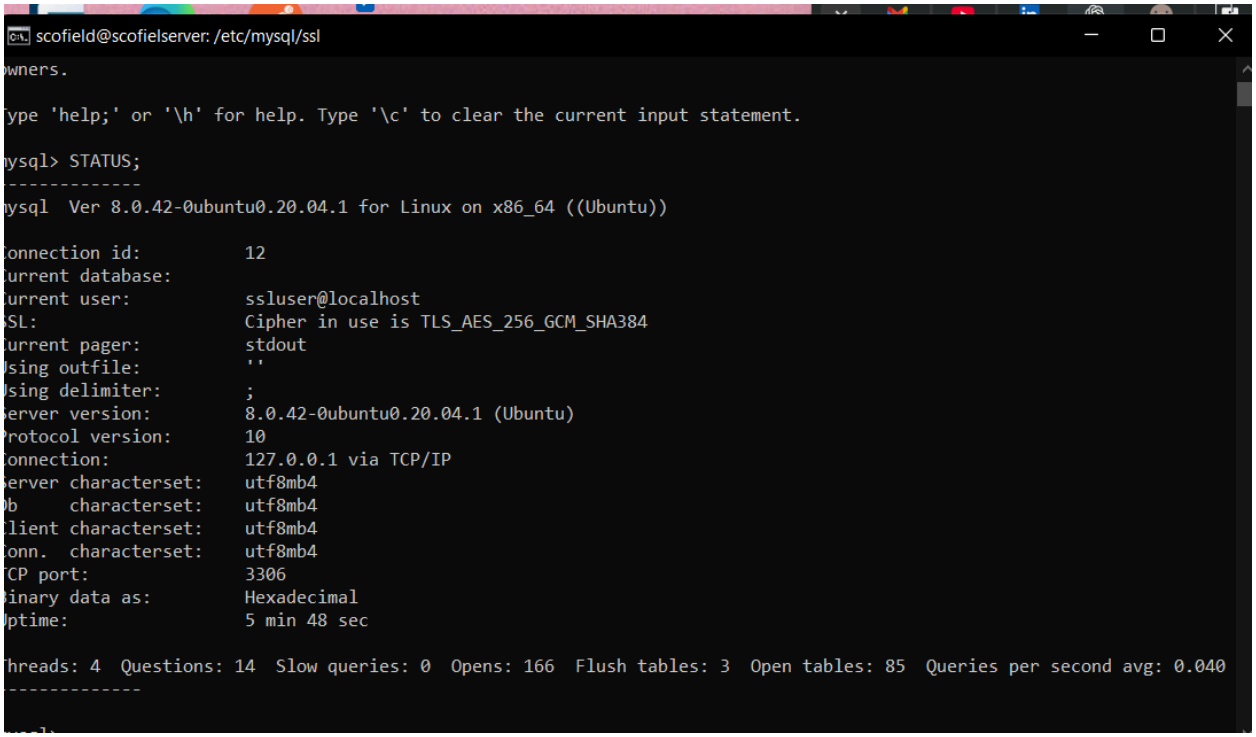
And added:

ssl-ca=/etc/mysql/ssl/ca.pem
ssl-cert=/etc/mysql/ssl/server-cert.pem
ssl-key=/etc/mysql/ssl/server-key.pem
require_secure_transport = ON

Then restarted MySQL:

sudo systemctl restart mysql

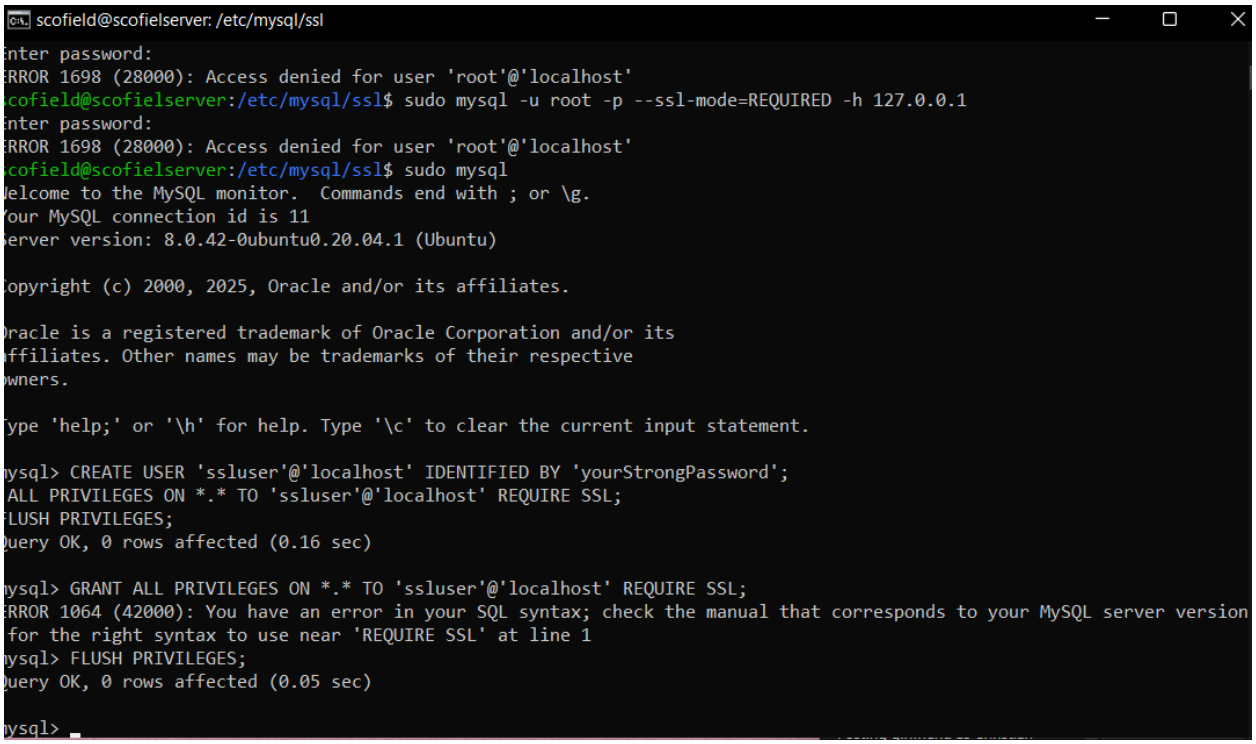
Screenshot: mysqld.cnf file with SSL lines



4. Creating an SSL-Required User

CREATE USER 'ssluser'@'localhost' IDENTIFIED BY 'myStrongPassword';
GRANT ALL PRIVILEGES ON *.* TO 'ssluser'@'localhost' REQUIRE SSL;
FLUSH PRIVILEGES;

This user is forced to use SSL for any connection.



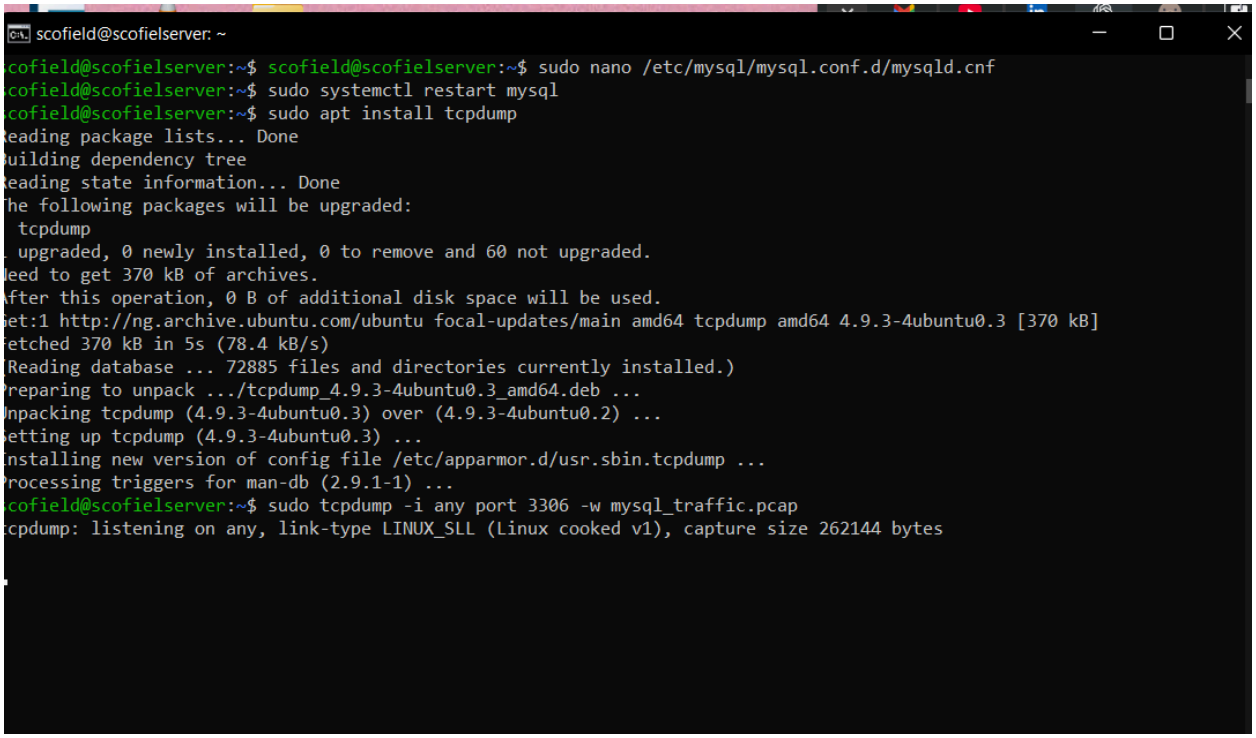
5. Testing the Connection

I tested the secure connection with:

mysql -u ssluser -p --ssl-mode=REQUIRED -h 127.0.0.1

Initially, I got an error saying **"server doesn't support SSL"**, which I fixed by properly generating the certificates and restarting MySQL after adding them to the config file.

Screenshot: SSL-enabled connection success



Inside the session, I ran:

CREATE DATABASE test;
USE test;
CREATE TABLE secure_data (id INT, name VARCHAR(100));
INSERT INTO secure_data VALUES (1, 'Scofield'), (2, 'TestUser');
SELECT * FROM secure_data;

---

6. Capturing Traffic

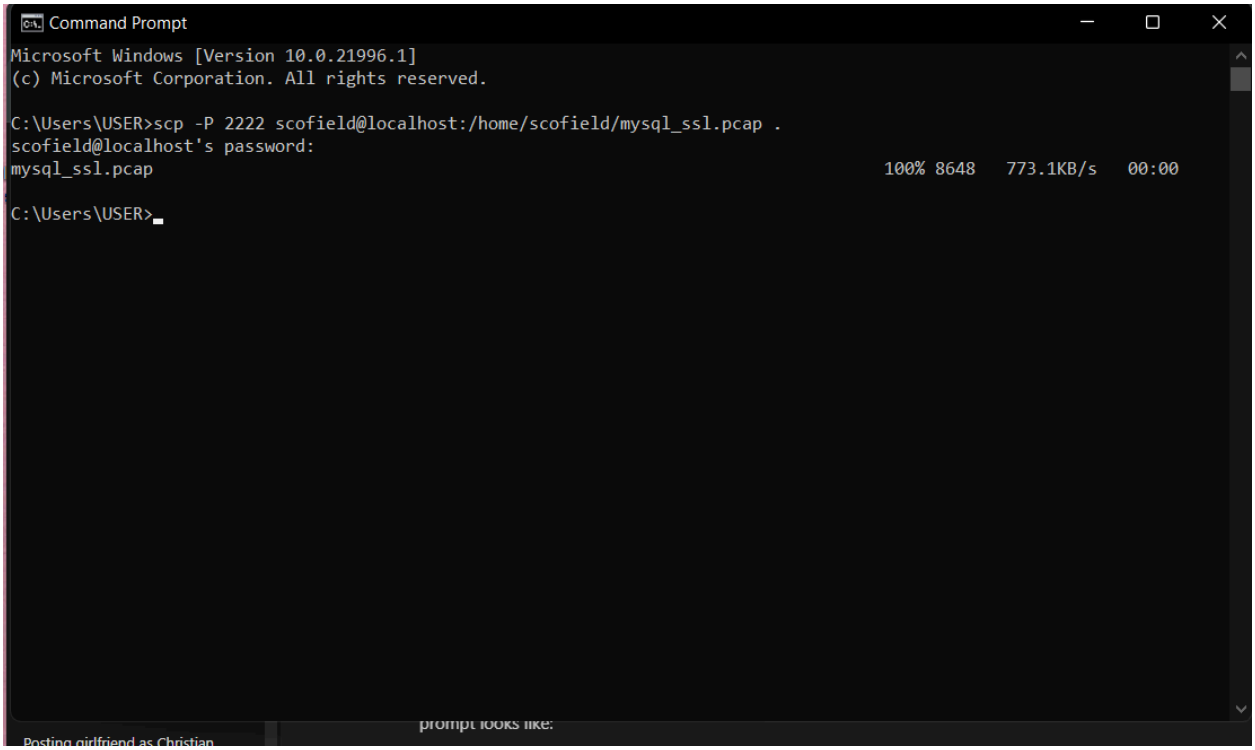
In a new terminal, I ran:

sudo tcpdump -i any port 3306 -w mysql_ssl.pcap

While the capture was running, I repeated the MySQL commands above using the `ssluser` account.

Then I stopped tcpdump with `Ctrl + C`.

Screenshot: tcpdump capturing traffic



7. Copying the `.pcap` File to Windows

This was tricky. I initially ran the `scp` command from inside the VM and kept getting errors like:

ssh: connect to host localhost port 2222: Connection refused
2222: No such file or directory

Eventually, I realized I needed to run the command from **Windows CMD**, not inside the VM.

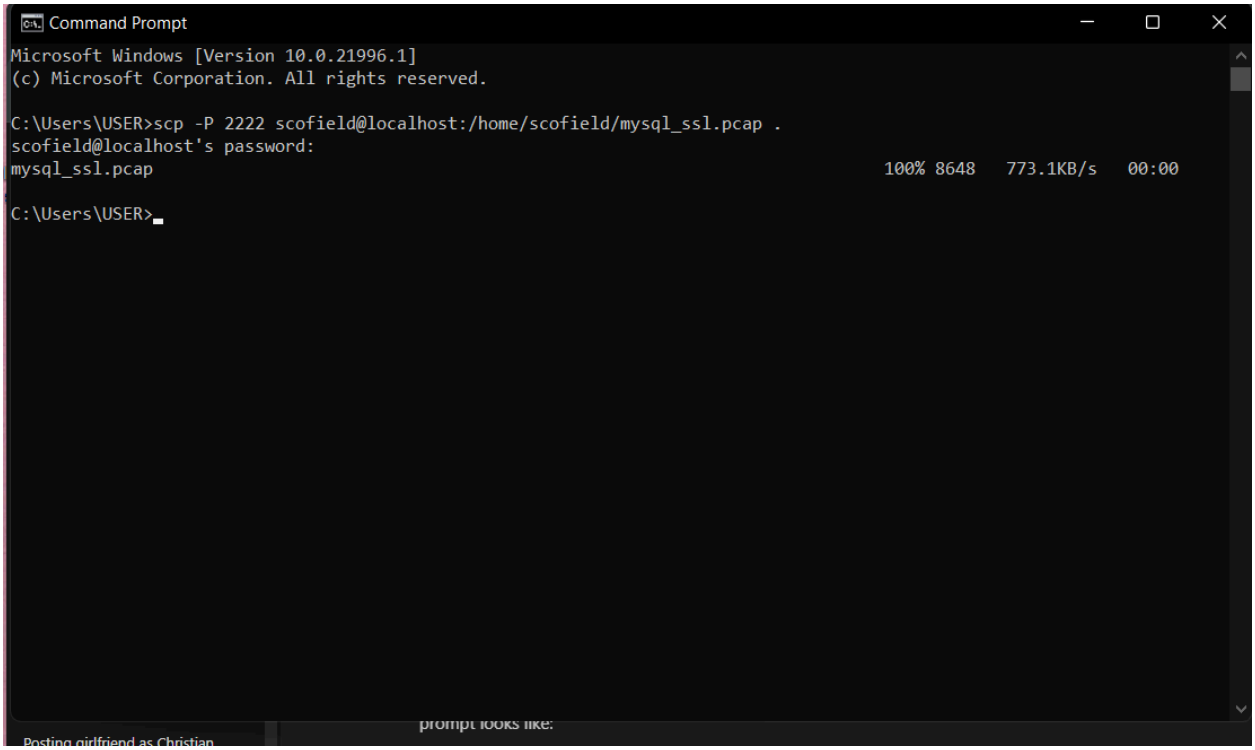
Correct command (run from Windows):

scp -P 2222 scofield@localhost:/home/scofield/mysql_ssl.pcap .

I had to make sure:

- Port forwarding was enabled in VirtualBox (Host: 2222 → Guest: 22)
- The file path was correct (`/home/scofield/mysql_ssl.pcap`)

After that, the file transferred successfully.



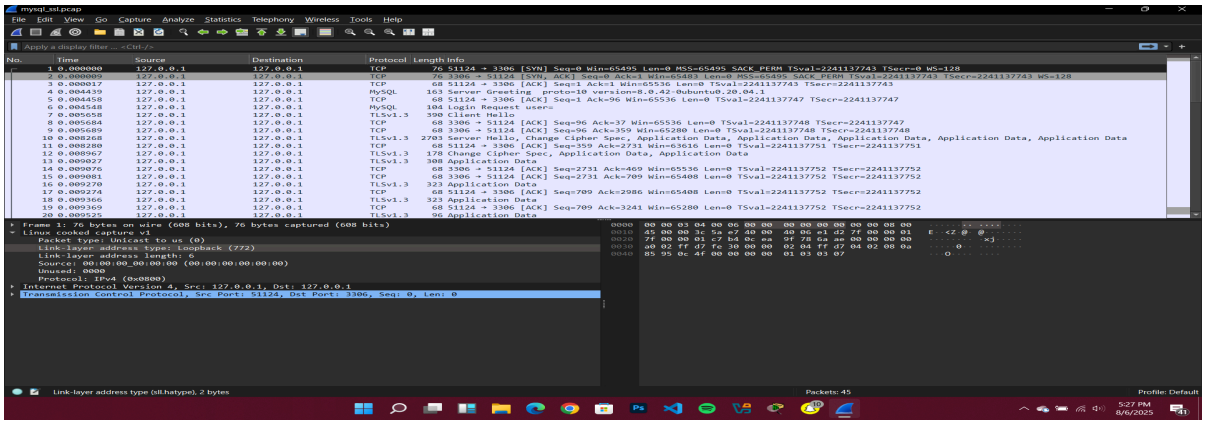
8. Analyzing in Wireshark

Opened the file in Wireshark and filtered for:

tcp.port == 3306

I saw a successful TLS handshake (`Client Hello`, `Server Hello`, `Certificate`, etc.) and all subsequent traffic appeared as **"Encrypted Application Data"**, confirming that the credentials and queries were protected.

Screenshot: TLS handshake and encrypted packets in Wireshark



Conclusion (for Q5)

- The TLS setup was successful.
- Passwords and data were not visible in plaintext.
- TLS handshake and encryption worked as expected.
- Key mistake: trying to use `scp` from inside the VM instead of from the Windows host.