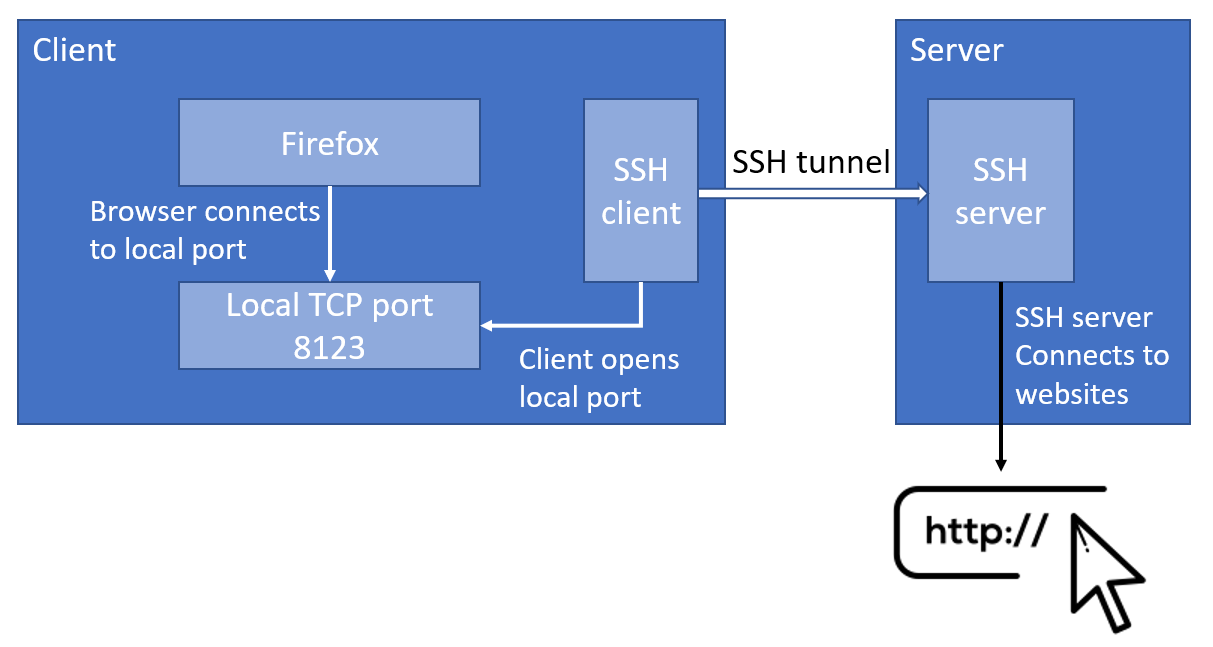
# SSH tunnels

## Requirements

Three components will be used in this workshop.

* An SSH server.  
  This server will be provided (Oracle Cloud Infrastructure compute instance)
* An SSH client.  
  The SSH client will be used to create an SSH tunnel and function as local SOCKS proxy server
* A browser.  
  The browser will use the local SOCKS proxy server to access the internet via the SSH tunnel

Graphically the setup will look as followed:



### Server environment

For this workshop, a remote SSH server and accounts are provided. The accounts are only available during this workshop.

Use SSH server: 132.145.250.238 port 443

User accounts: wsuser1 to wsuser30

Passwords: WsusAMIS\_1 to WsusAMIS\_30 (number corresponds to username)

Do you want to try this out for yourself, create your own always free OCI instance at cloud.oracle.com. A description on how to do this is provided at:

<https://technology.amis.nl/2020/02/23/secure-browsing-using-a-local-socks-proxy-server-on-desktop-or-mobile-and-an-always-free-oci-compute-instance-as-ssh-server/>

### SSH client

You can choose from several tools to use as SSH client and SOCKS proxy server, depending on which you want to try.

#### Mac OS / Linux / UNIX

The regular ssh client (preinstalled on all known Linux, Unix, Mac OS distributions) will be used to configure the SSH account and create a tunnel + SOCKS proxy server.

For Linux and Unix you need to have an SSH client installed. Usually it already is (such as for Mac OS). If not, install it first.

Debian, Ubuntu and similar: sudo apt install openssh-client

Red Hat, Fedora and similar: sudo yum install openssh-clients

#### Windows

MobaXterm will be used as SSH client to configure the server account and software to configure the tunnel / SOCKS proxy server. You can download it at <https://mobaxterm.mobatek.net/>. After downloading, install it.

If you’re using Chocolatey (<https://chocolatey.org/>), you can install MobaXterm, from a console which has Admin privileges, using:

choco install mobaxterm

#### Android

For Android phones you can use ConnectBot as SSH client (although typing on a mobile phone can be challenging) and as SSH tunnel configuration tool / SOCKS proxy server. ConnectBot can be downloaded and installed from the Google playstore here:

<https://play.google.com/store/apps/details?id=org.connectbot&hl=en>

It is not recommended to use ConnectBot to generate the SSH public and private keys and to add them to the authorized\_keys file on the server. Typing, copying and pasting can be challenging on a mobile device. You should use ConnectBot though to configure the SSH client to login using the generated private key and to create the SOCKS proxy running on your mobile device.

### Firefox browser

You need a browser to use the SOCKS proxy server. Firefox is available for Linux, Mac, Android and can be configured to use a SOCKS proxy server. In this workshop Firefox will be used. You can download it here: <https://www.mozilla.org/en-US/firefox>

## Testing connectivity

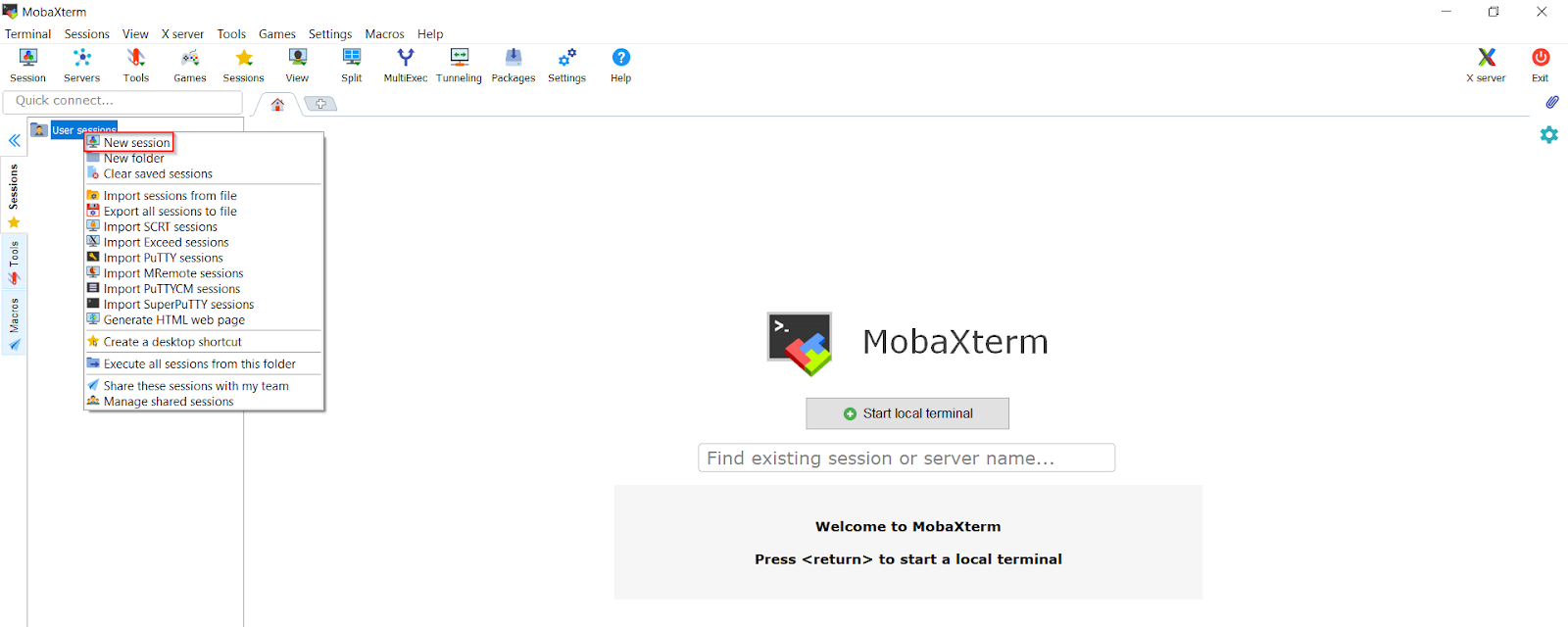
Use your preferred SSH client to test connectivity to the SSH server and login. Remember to use the following credentials:

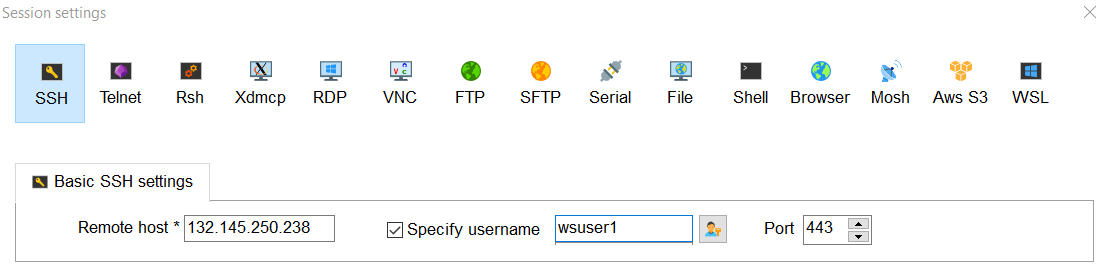
Use SSH server: 132.145.250.238 port 443

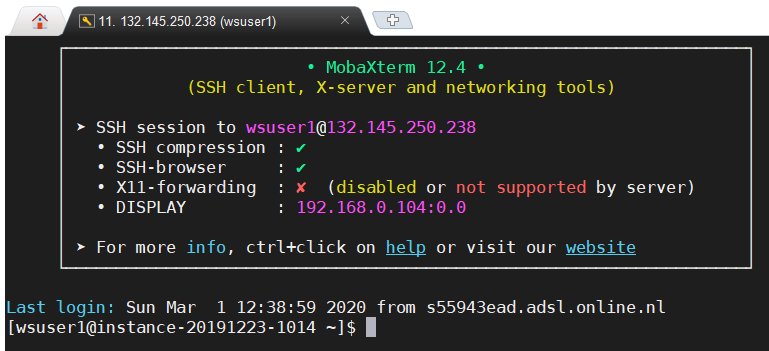
User accounts: wsuser1 to wsuser30

Passwords: WsusAMIS\_1 to WsusAMIS\_30 (number corresponds to username)

### MobaXterm







### CLI: Linux, Mac

ssh wsuser1@132.145.250.238 -p 443

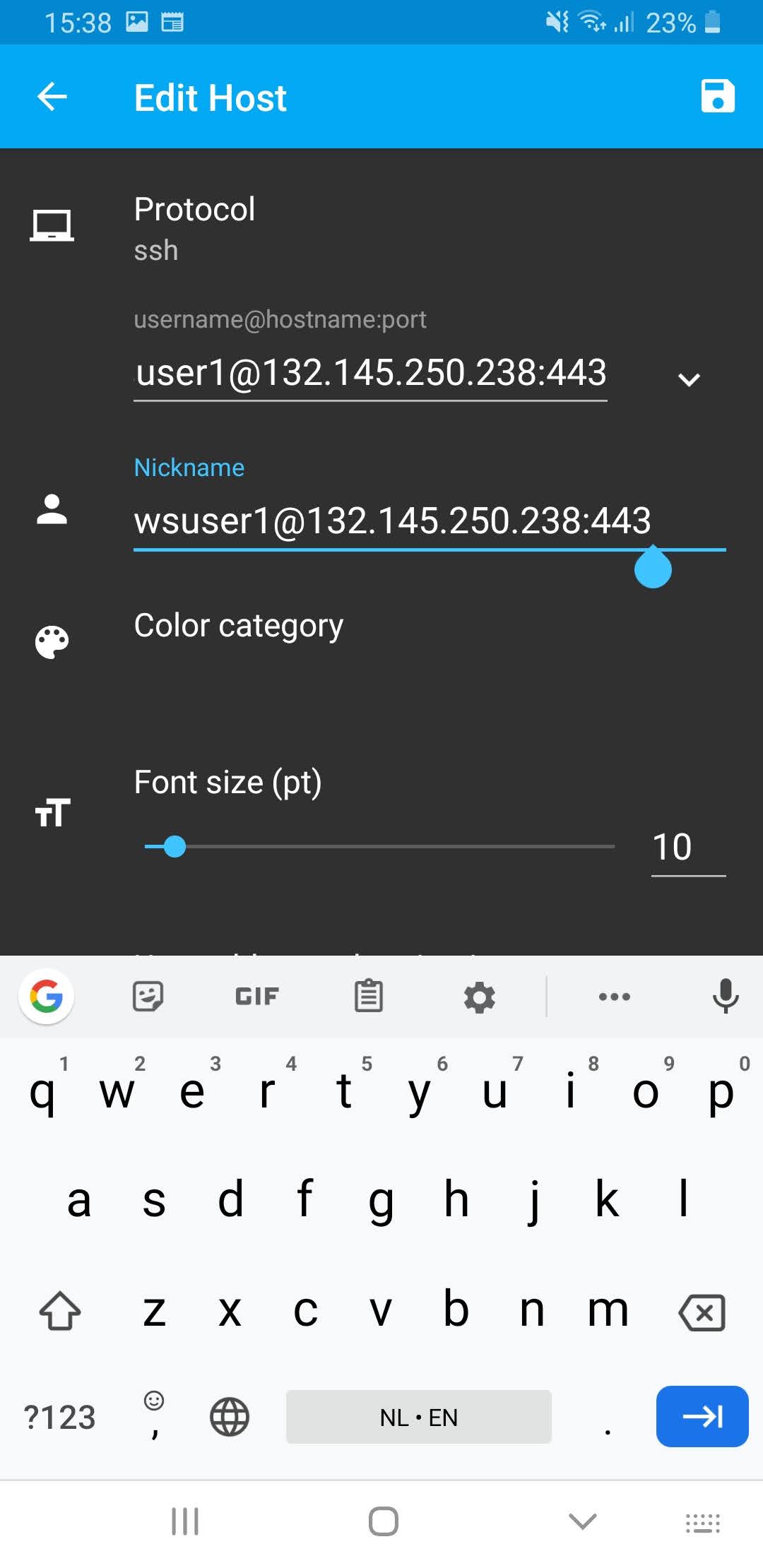
### Android: ConnectBot

Start ConnectBot. Click the + icon at the bottom of the screen.

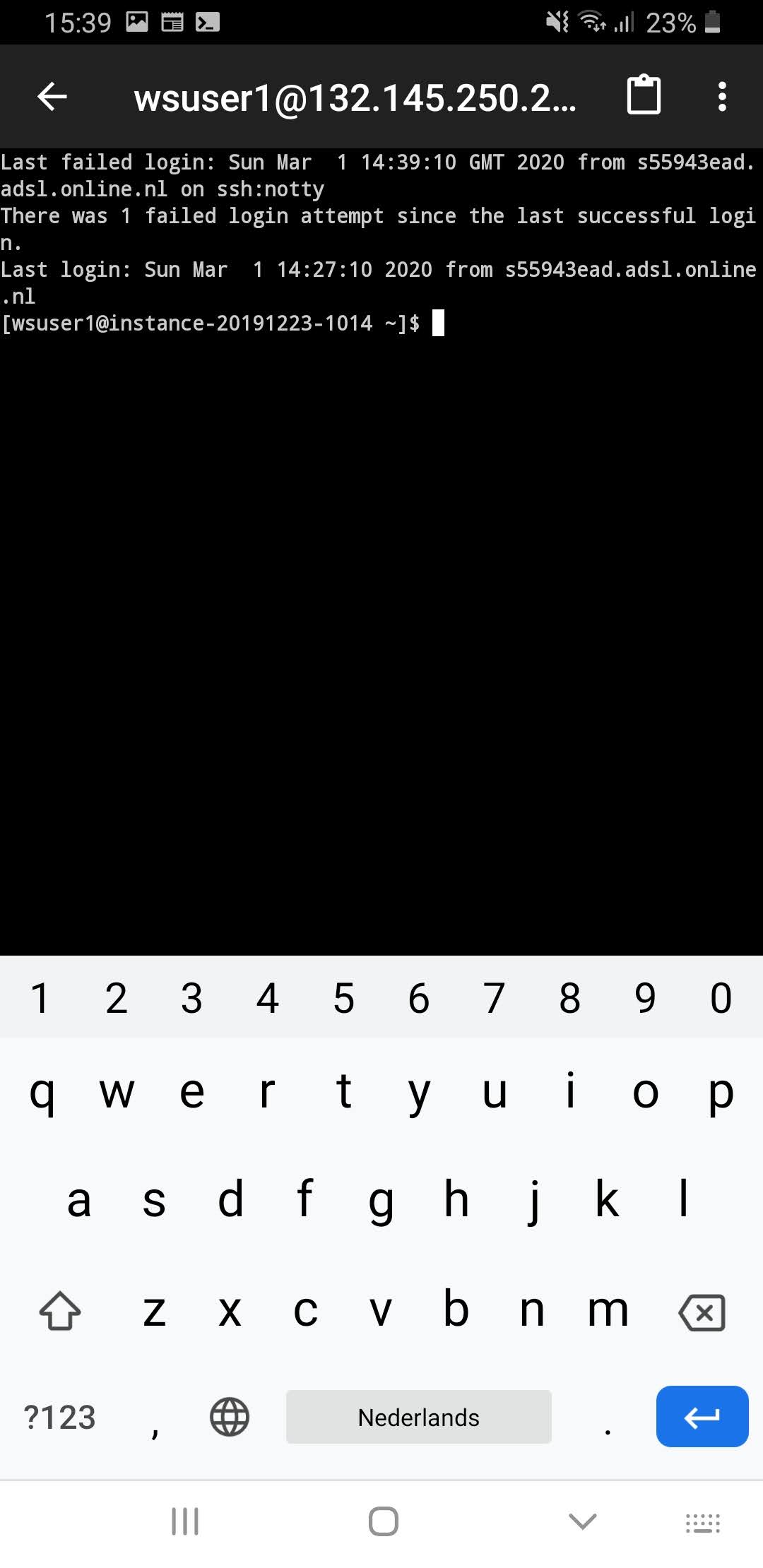


Input the following properties (replace the wsuser with your assigned user):

username@hostname:port wsuser1@132.145.250.238:443



Connect to the newly created host by tapping it. Type your password.



Confirm you are connected.

## Generating a public and private key

In order to login to the SSH server, a private key will be used. A public key needs to be present in authorized\_keys file on the SSH server.

First we will generate a public and private key pair to use. Next we will login using a username/password to the SSH server and register the public key.

You can choose to do this by command line or by using a GUI. You do not need to do both. You can use the terminal from the previous section to generate the public and private key file from the command line.

### CLI: Linux/Unix/Mac or inside a MobaXterm terminal

Start a terminal (or use the one already open from the previous section)

ssh-keygen -t rsa -m PEM -C "youremail@email.com" -f rsakey

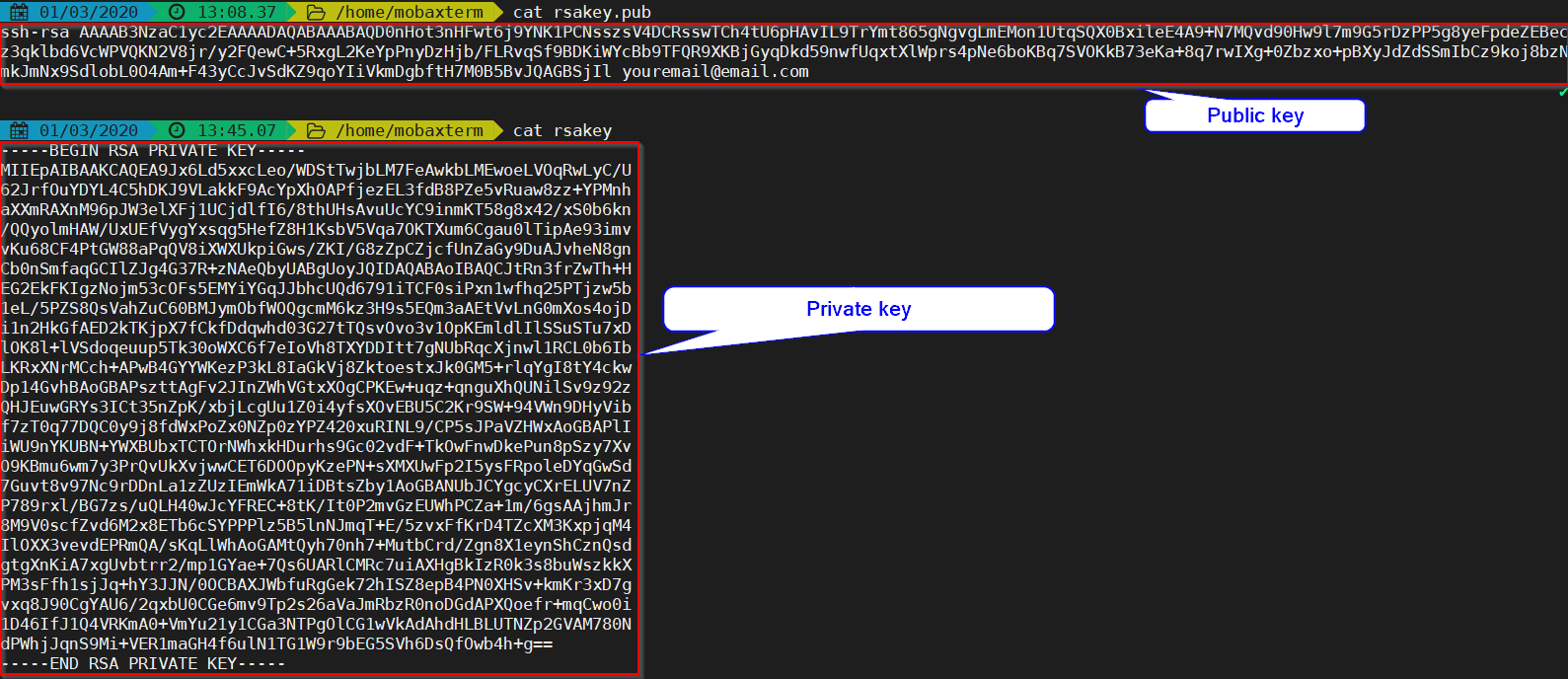
Press enter twice (no keyphrase)

This generates two files in the current directory:

Private key: rsakey

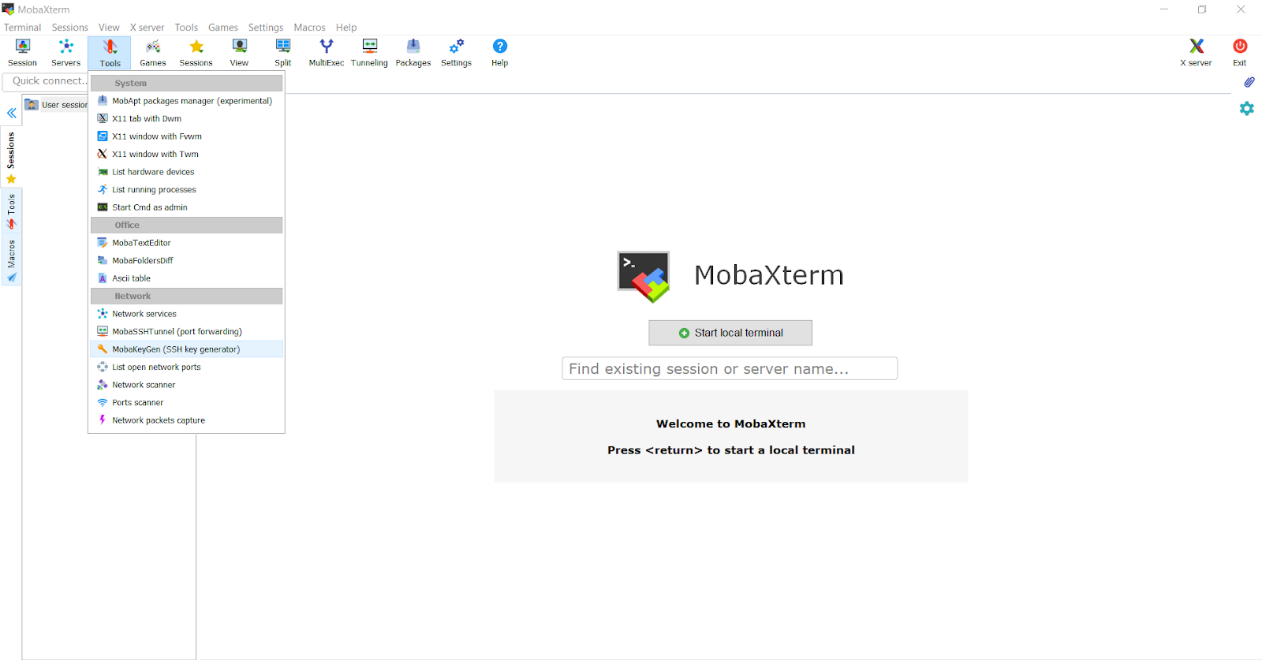
Public key: rsakey.pub

rsakey is the private key used to login. rsakey.pub is the public key which will be registered on the SSH server. You can copy the public and private key for future usage (it will be referred to in later steps).

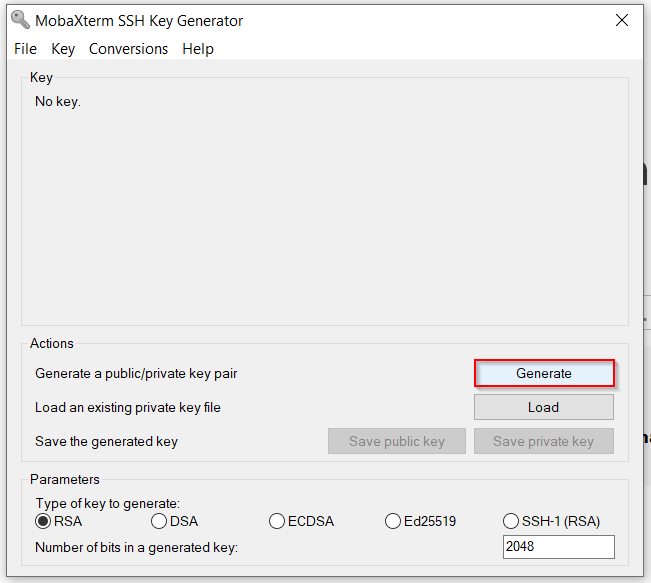


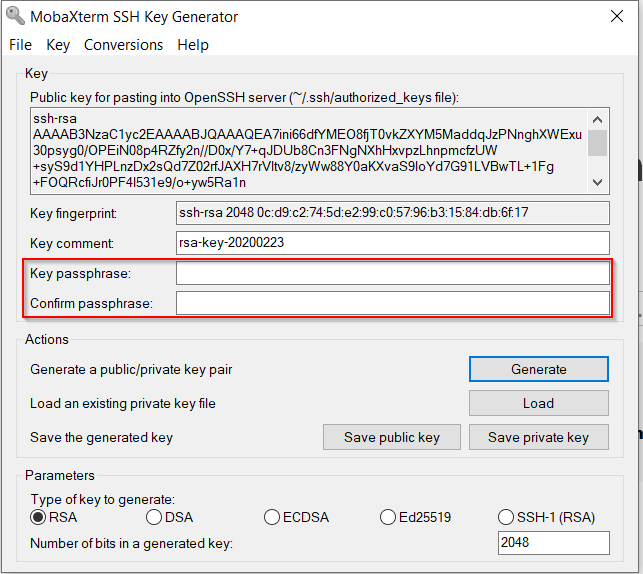
### GUI: MobaXterm / PuTTYgen

An alternative to executing shell commands to generate a keypair is by using tools like MobaXterm / PuTTYgen. PuTTYgen is available for Windows, Mac and Linux. MobaXterm is only available for Windows.

From MobaXterm you can start the MobaXterm SSH Key Generator. 

First generate a keypair





Mind that when saving the public and private key, not to supply a passphrase. Not every client can deal with that.

Copy the key as displayed in the ‘Public key for pasting into OpenSSH server (~/.ssh/authorized\_keys)’ and save it in a separate file for later usage. For example openssh\_key.key

Save the public key and private keys to separate files. For example pubkey and privkey. You will use the privkey to login.

## Add the public key

You have previously copied a public key which looks something like:

ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEAqtGsoOOTD9A3pPMDbEijYSxO375SrV1s25bkFXs7U2WLWKQcr/ApafOVeWvocjr+ZSuDzzD4f9VT7wfmb8LWm4yurDFWKdSEJRujEBndpTJDtBnboJYvZoSz6A3An8vRyxTjwqDQhZURiVMEt0D40WJBy64Mu25x2LHIneNfL5h6wP4nGQ4AD+OjbOUEd4OjTaEUx+YWHZkqNj4aQ091SqdYuaokYeUgzkub9HMKTxDB7OQOoFTN5GKiXGZtnl4exGEcfCSqZd8rnmo6YF++gcsseJabtaQ+GznPs4AiDoaX9r3F1UoARFwMMNN4APejmCBNkGdjCi+7ESmROyMTXQ== rsa-key-20200301

This public key will be added to the ~/.ssh/authorized\_keys file.

Execute the following commands (using a CLI) to create the required folders, the file and set the correct permissions:

mkdir ~/.ssh

chmod 700 ~/.ssh

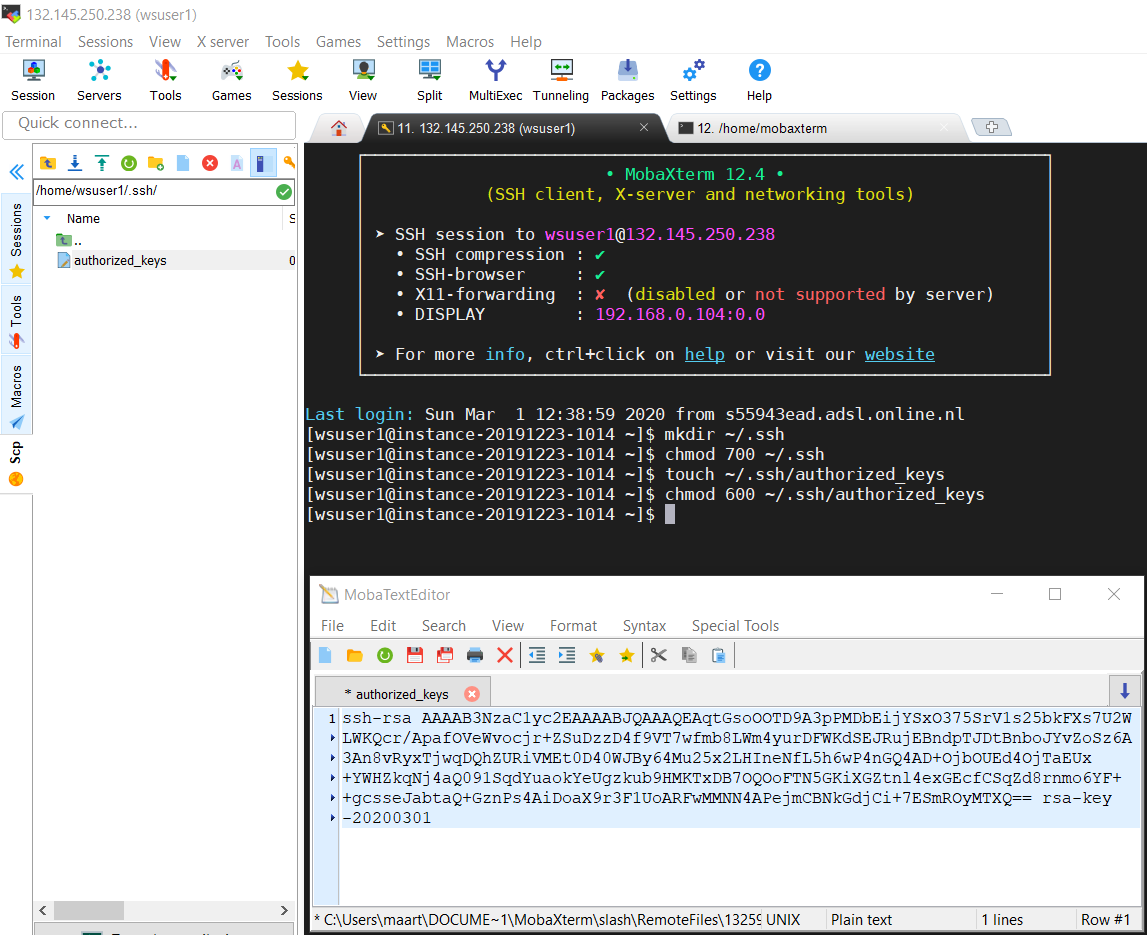
touch ~/.ssh/authorized\_keys

chmod 600 ~/.ssh/authorized\_keys

Now add the public key to the authorized\_keys file. You can do this in multiple ways

### MobaXterm (Windows)

After the authorized\_keys file is created, you can edit it with the MobaTextEditor by opening it from the Scp tab on the left. If you cannot immediately see the .ssh folder, click the refresh button or press F5. Paste the public key into this file and save it (the red disk icon). Confirm you want to overwrite the file on the server.



### CLI: Linux, Mac

If you have generated the public and private key on the provided SSH server, the easiest way to add the key is

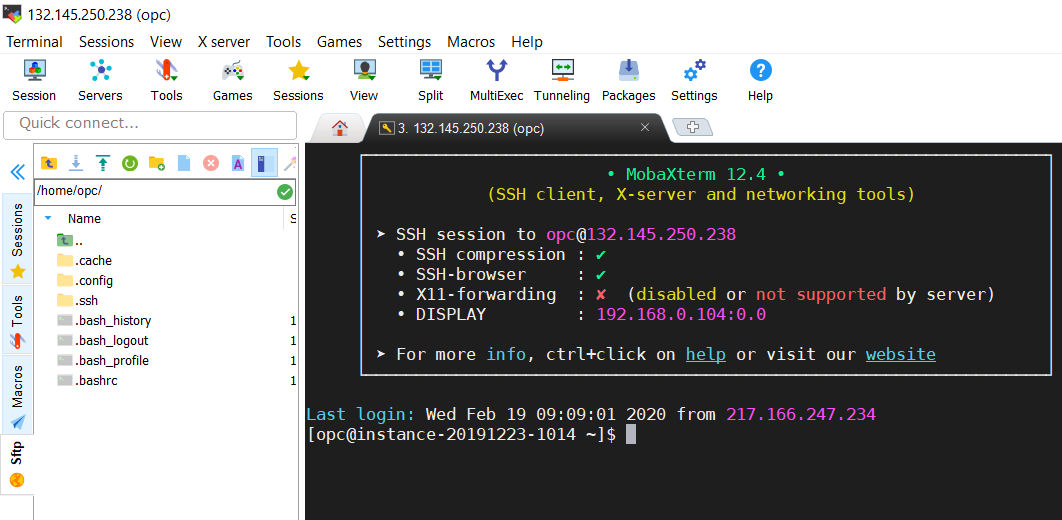
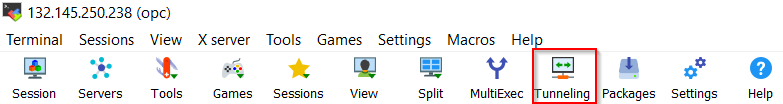
cat ~/rsakey.pub >> ~/.ssh/authorized\_keys

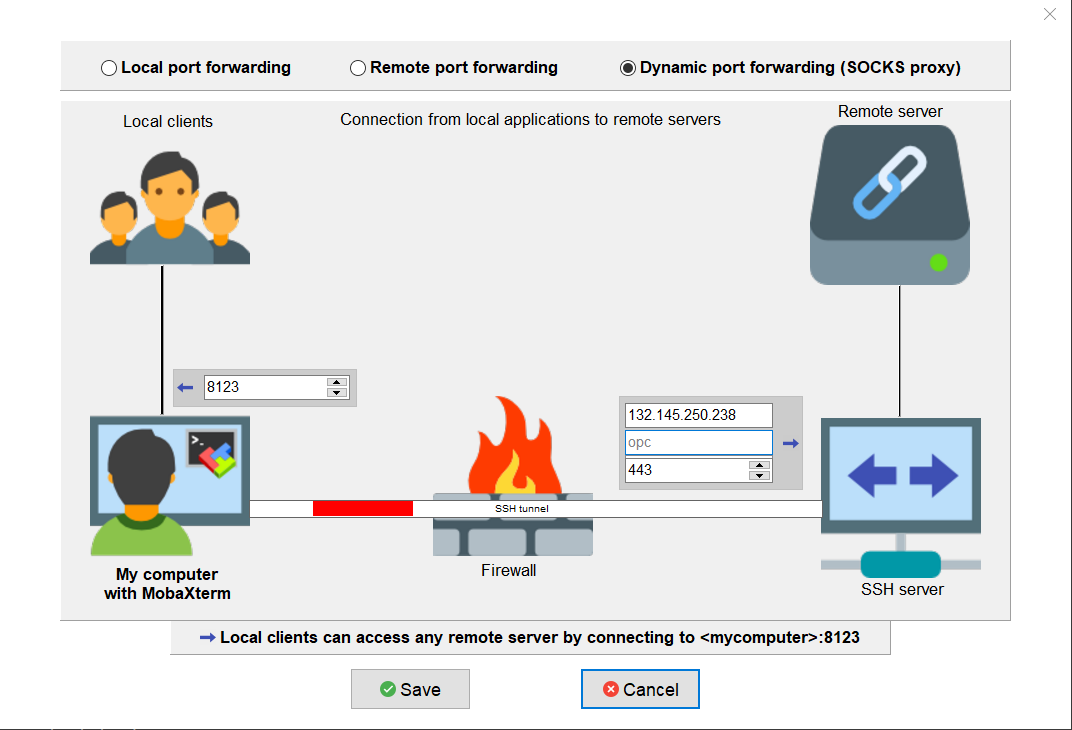
Or edit ~/.ssh/authorized\_keys with vi (or any other editor) and add the public key. Make sure you add the public key to the remote account! (this might differ from the location where you generated the keypair)

## Creating an SSH tunnel / SOCKS proxy server

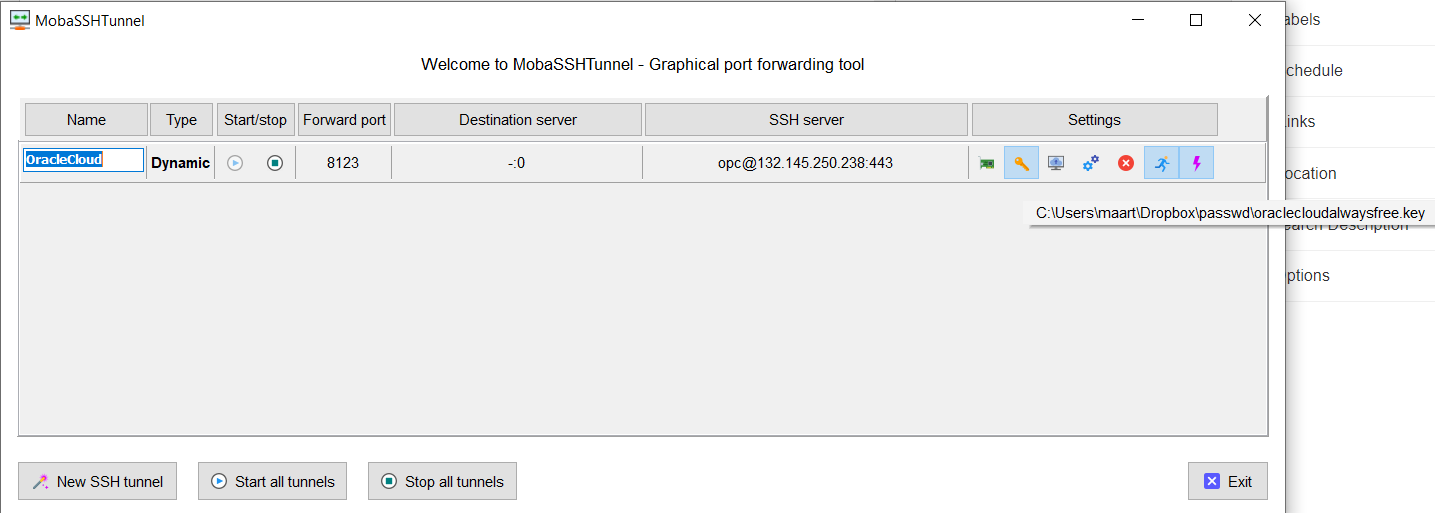
### MobaXterm (Windows)

MobaXterm provides a nice interface to configure a tunnel with. First start the interface.



Next configure the connection details as indicated below. Replace opc with your wsuser

Configure the private key to use and start the tunnel / SOCKS proxy server.



### CLI (UNIX, Linux, Mac)

Using the SSH command you can create an SSH tunnel / local proxy server

nohup ssh -i ~/oraclecloudalwaysfree.key -D 8123 -f -C -v -N opc@132.145.250.238 -p 443

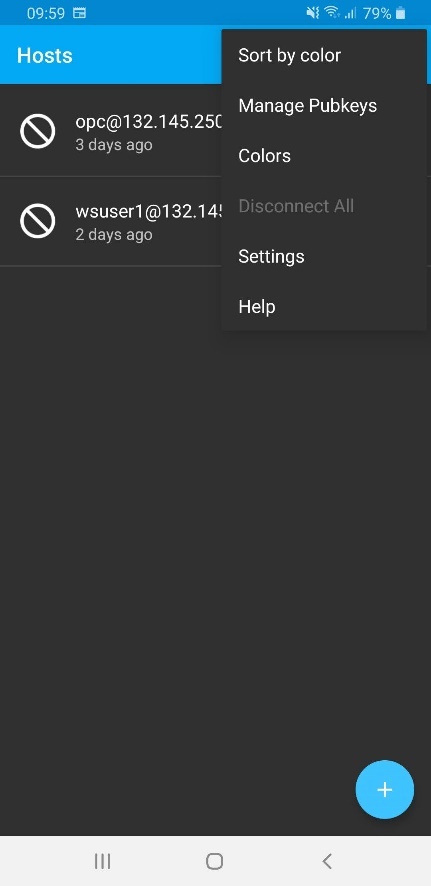
Here also replace opc with your wsuser.

* -D 8123 starts a SOCKS 4 and SOCKS 5 compliant proxy server on port 8123
* -i indicates the private key to use
* -f indicates background execution of SSH
* -C requests compression of data
* -v gives verbose output. Useful for debugging
* -N indicates no remote command needs to be executed. we just need the tunnel functionality
* -p indicates the port to connect to on the remote host.
* opc@132.145.250.238 indicates the user and host to connect to

You can monitor the proxy server by checking out ~/nohup.out

### ConnectBot (Android)

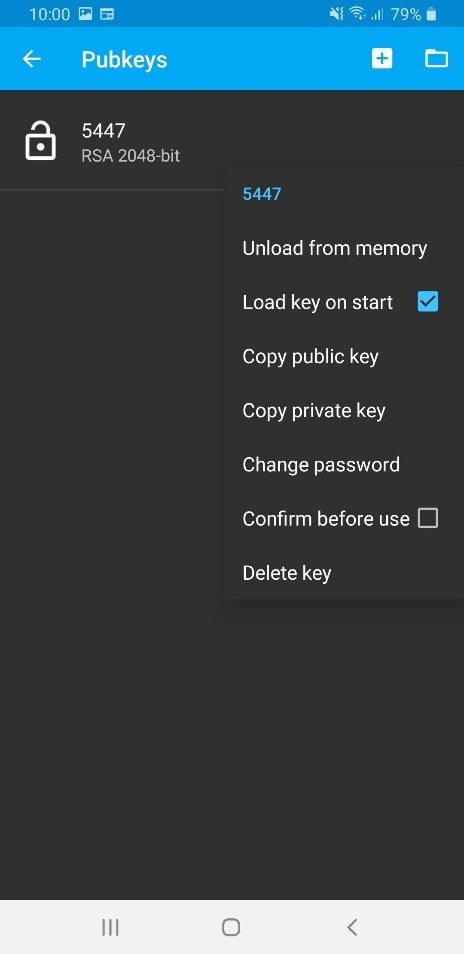
Add the private key. Click Manage Pubkeys.



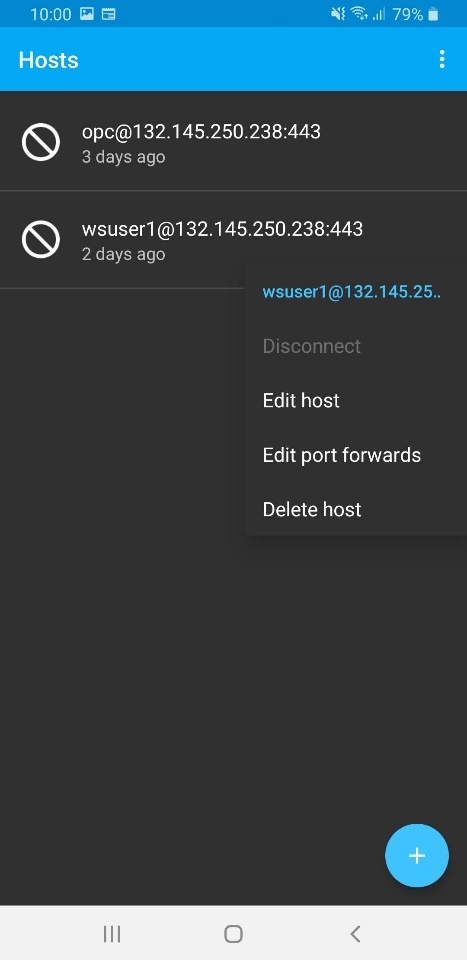
Select your private key. For getting the key to your mobile phone, you can mail it to yourself, download it from the mail and select it from the Downloads folder.



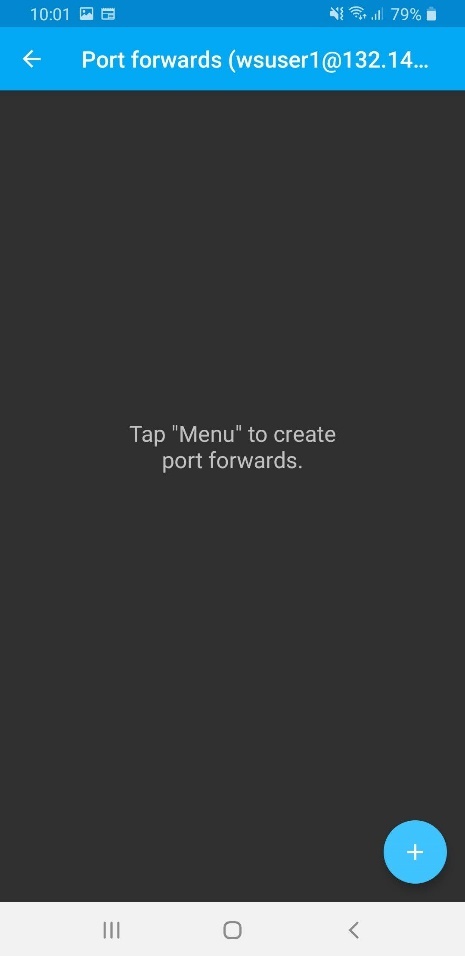
Configure the key to load on startup and unlock the key

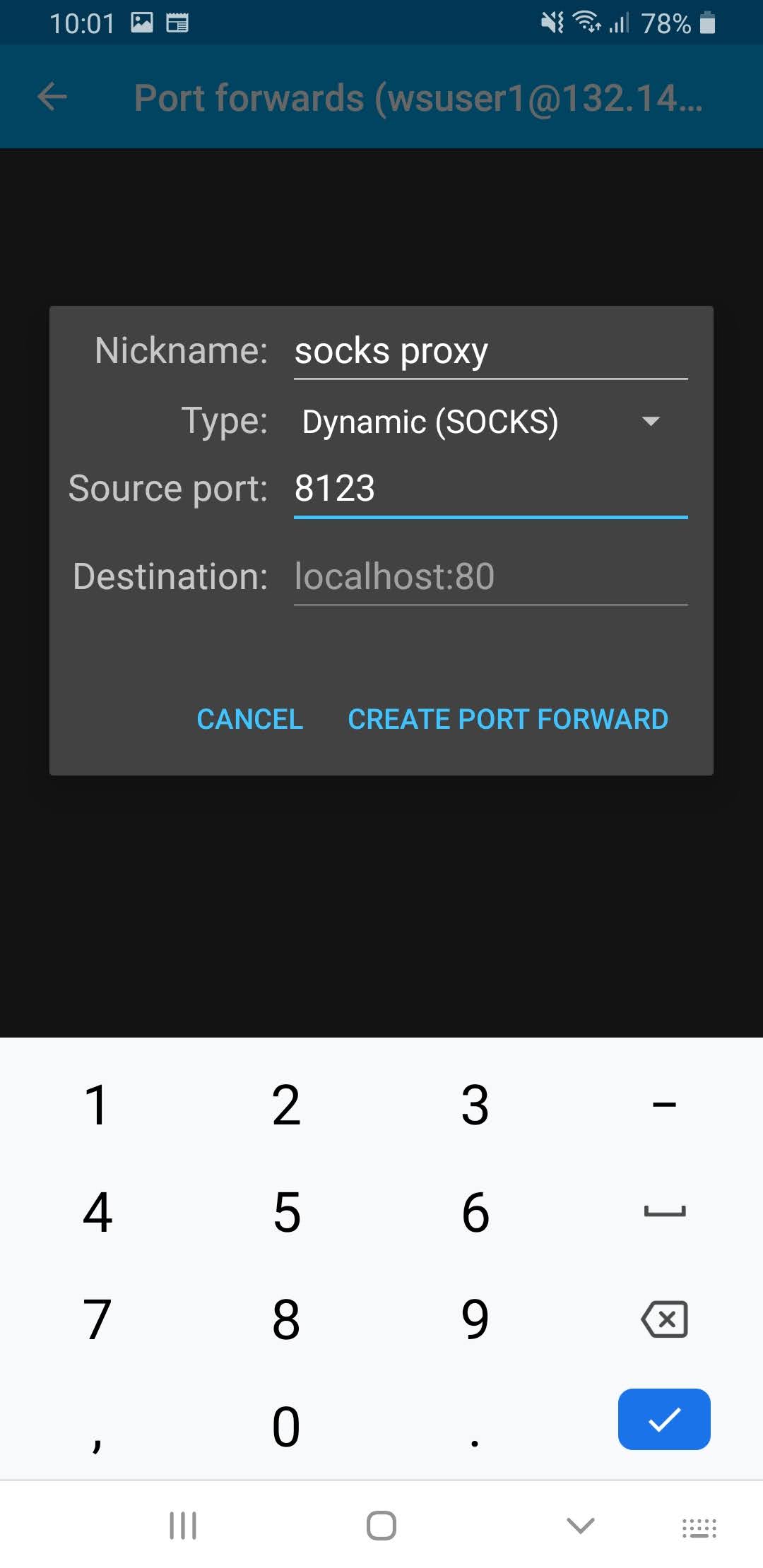


Configure port forwards



Add a new port forward and use the settings as described below. A dynamic SOCKS proxy on port 8123

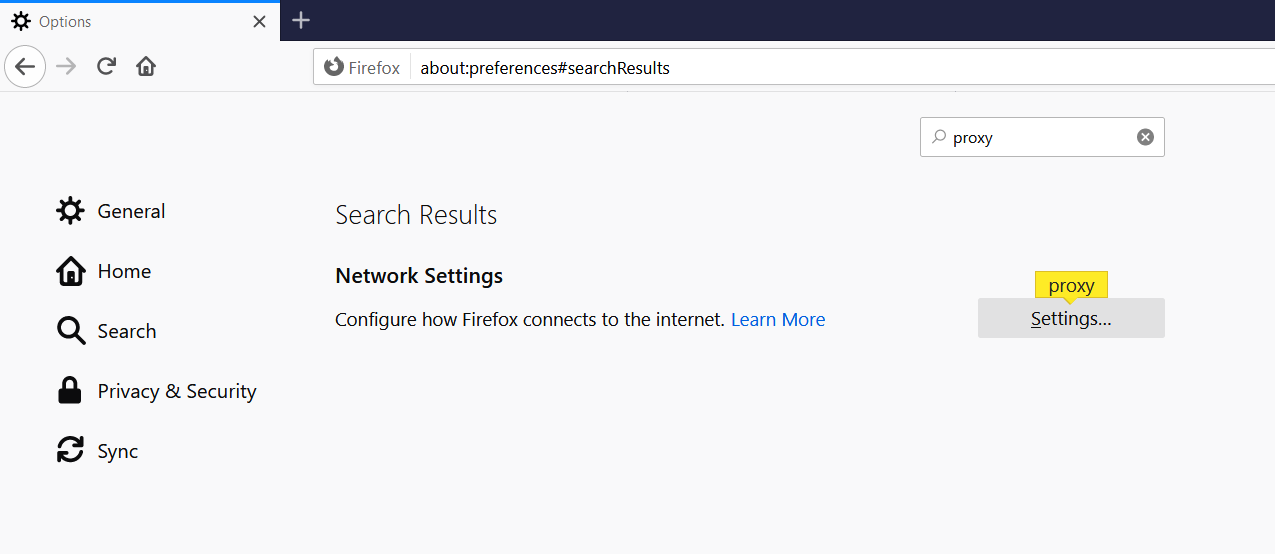


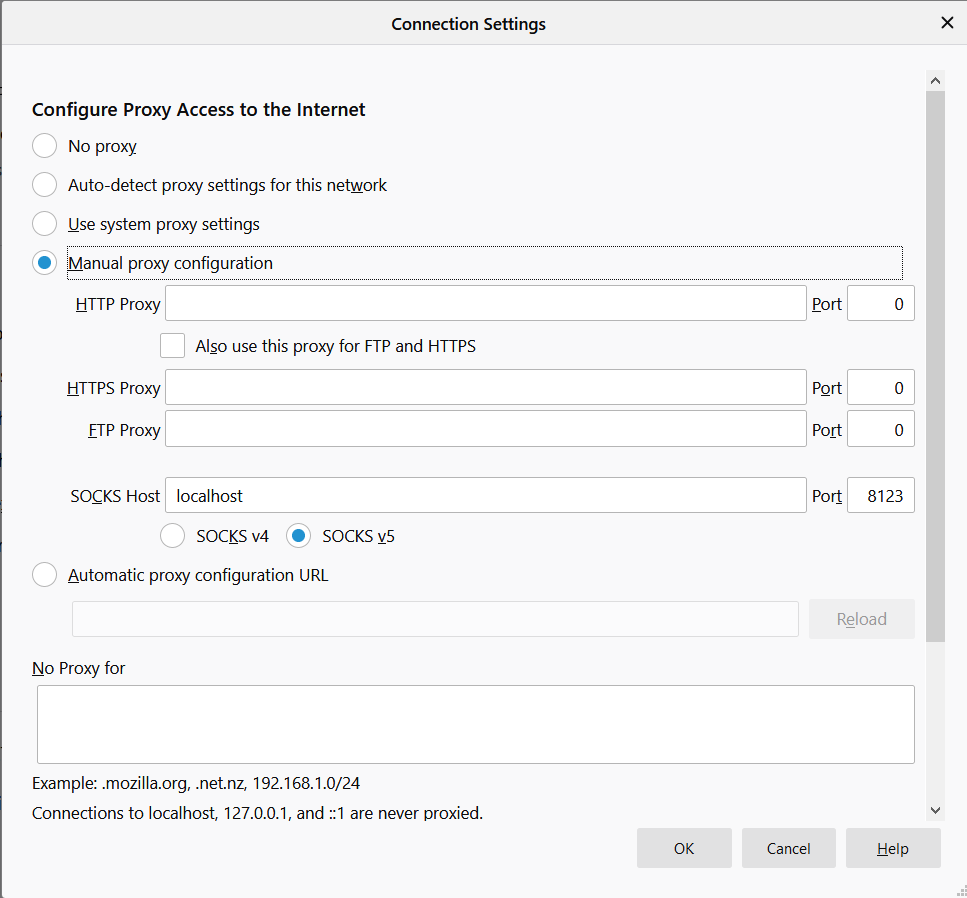


## Configuring clients

### Firefox desktop (Linux, Windows, Mac)

Open the Firefox preferences and search for proxy. Open the Network settings



Configure the SOCKS proxy server to use. localhost:8123

### Firefox mobile (Android)

For Firefox on a mobile device this is slightly harder, but on for example Chrome, these settings are not available at all. In Firefox the same settings as described above are available but not nicely from a GUI. The following here describes the steps you need to take.

In the firefox URL bar, type ‘about:config’ and press enter to access advanced settings. Search for ‘socks’ and set the following settings:

* network.proxy.socks = 127.0.0.1
* network.proxy.socks\_port = 8123
* network.proxy.socks\_remote\_dns = true

Search for ‘proxy.type’ and set the following setting:

* network.proxy.type = 1

## Access the web using a remote server

Now confirm you can access the web using your OCI instance by going to [www.whatismyip.com](http://www.whatismyip.com) and check that Oracle Public Cloud is your ISP.

