Remote Access to your Workstation

For security reasons, access to your Lorentz workstation is only possible within the Lorentz Institute intranet. Remote access can occur either securing your connection via an intermediate step called *SSH tunneling* (AKA *port forwarding*), through the Lorentz Institute VPN service or via the Lorentz Institute Remote Workspace.

Following are some examples that demonstrate the concept of SSH tunnelling. For alternative methods of connection, please see the relevant documentation. SSH access to our servers requires you to set up two-factor authentication (2FA) on your account for security reasons.

 The examples below have been tested with OpenSSH v7.3+.

SSH tunneling

By means of an SSH tunnel you can transport any arbitrary data over an encrypted SSH connection. Members of the Lorentz Institute can use this technique to gain remote shell access to their workstation across our firewall which would prevent access otherwise.

How does it work?

You must have an ssh client installed on your personal device – e.g. laptop, PC – in order to establish a *tunnelled* connection.
The Lorentz Institute has a dedicated server (SSH server) ready to listen to any (authenticated) client connections.
Once a client-server connection is established, a given application contacts the SSH client on a chosen port on which the client is listening.
The SSH client in turns forwards all encrypted application data to the server which finally communicates with the actual application server.

For remote ssh connections to your IL workstation, the steps above can be summarised into the following. Establish an ssh client-server to our SSH server and instruct your SSH client to forward any new SSH-connection data that will be sent to an arbitrary port number to go via our SSH server. The server will then relay this information to the SSH server running on your workstation.

Example 1

Establish an SSH connection to `workstation.lorentz.leidenuniv.nl` via our SSH server `styx.lorentz.leidenuniv.nl`

```
ssh -o ProxyCommand="ssh -W %h:%p username@styx.lorentz.leidenuniv.nl"
username@workstation.lorentz.leidenuniv.nl
```

For connections that will use the DISPLAY environment variable (think of any application with a GUI), add the option `-X` to your SSH commands.

Example 2

As in *Example 1* but this time using your client ssh configuration file usually located at `$HOME/.ssh/config` on GNU/Linux systems

```
# cat $HOME/.ssh/config
Host workstation.lorentz.leidenuniv.nl workstation
     ProxyCommand /usr/bin/ssh -W %h:%p styx.lorentz.leidenuniv.nl
     User username
```

Once this configuration is in place, a simple `ssh workstation` will get you to your workstation.

Example 3

Establish a web browser connection to a jupyter notebook on `workstation.lorentz.leidenuniv.nl` port YYYY.

Configure your local `$HOME/.ssh/config` as below

```
Host styx
   HostName styx.lorentz.leidenuniv.nl
   LocalForward YYYY localhost:YYYY

Host workstation
   HostName workstation.lorentz.leidenuniv.nl
   ProxyJump styx
   LocalForward YYYY localhost:YYYY
```

Browse to `http://localhost:YYYY`.

Example 4

Establish a web browser connection to a Jupyter Notebook session running on node marisXX when outside the IL intranet [1)]

```
Host lorentz
     HostName ssh.lorentz.leidenuniv.nl
     User username

Host maris
     HostName xmaris.lorentz.leidenuniv.nl
```

```
     ProxyJump lorentz
     User username

Host marisXX
     HostName marisXX.lorentz.leidenuniv.nl
     ProxyJump maris
     User username
     LocalForward YYYY localhost:YYYY
```

Browse to http://localhost:YYYY.

SSH access/tunnelling behind firewalls

There are situations in which SSH could be forbidden by firewall settings of the internet service provider. Think of countries which limit freedom of speech for example. Luckily Lorentz Institute provides its members with a special access server to overcome these restrictions.

In a nutshell, IL offers SSL-wrapped SSH access, that is it conceals SSH connections using the SSL protocol which is the protocol used by the world wide web to serve `https` connections. In other words, to the eavesdropper your SSH connection will just look like a normal web connection.

The set up on your side is rather simple and requires only editing a file on the SSH client you wish to use, e.g. laptop, workstation, etc..

Add the following stanza to your SSH client config file[2] (`~/.ssh/config` on most GNU/Linux distros)

```
Host ssh.lorentz.firewall
  ProxyCommand openssl s_client -connect access.lorentz.leidenuniv.nl:443 -
servername lorentz -quiet
  User <Your IL username>
```

Then to initiate a SSL-wrapped SSH connection open a terminal and type

```
$ ssh ssh.lorentz.firewall
depth=2 C = US, ST = New Jersey, L = Jersey City, O = The USERTRUST Network,
CN = USERTrust RSA Certification Authority
verify return:1
depth=1 C = NL, O = GEANT Vereniging, CN = GEANT OV RSA CA 4
verify return:1
depth=0 C = NL, ST = Zuid-Holland, O = Universiteit Leiden, CN =
access.lorentz.leidenuniv.nl
verify return:1

---------------------------------------------------
    Welcome to the Lorentz Institute workstations
    Access is allowed for authorized users only.
    Any abuse will be tracked.

    Helpdesk     Room HL40[7-9]   Tel 8484
```

```
    https://helpdesk.lorentz.leidenuniv.nl
    support@lorentz.leidenuniv.nl
-------------------------------------------------

READ THIS CAREFULLY BEFORE PROCEEDING:
--------------------------------------
https://ilorentz.org/wiki/doku.php?id=institute_lorentz

Last login: Tue May 17 09:36:49 2022 from XX.XX.XX.XX
*****

$
```

When the connection is initiated you will be able to double-check the SSL certificate details, especially the CN entry (see above) which must correspond to our server `access.lorentz.leidenuniv.nl`. Then upon a successful authentication, you will be let in and be able to use the command line as usual.

Similarly it is possible to initiate an SSL-wrapped SSH SOCKS proxy connection useful to protect your browser sessions from eavesdroppers as in the example below. Provided you set up your SSH client config as described above, type

```
ssh -ND 8888 ssh.lorentz.firewall
```

then modify your browser settings to instruct it to redirect all connections to a SOCKS proxy listening on `localhost` post 8888.

1)
This method will only work if you have a slurm-controlled running jupyter session on marisXX. See xmaris.
You are strongly encouraged to use xmaris OOD facilities nonetheless.
2)
The same result is obtained by executing directly `ssh -o ProxyCommand="openssl s_client -quiet -connect access.lorentz.leidenuniv.nl:443" ssh.lorentz.firewall` on the command line.