1 Course VM

Each student is being given a virtual machine which is dedicated to this course. You may use it for any activities relevant to this course or your CS interests.

The VM can only be accessed using ssh from **within** the campus network. Off-campus access will require setting up a ssh tunnel as outlined at the end of this document.

The VM runs Ubuntu 18.04. Software which will be required by the course is pre-installed, but it is possible that you may need to install additional software during the course of the semester.

You should make sure that you perform each of the following steps:

1. Initial password change.
2. GUI access to your VM (`x2go` recommended).
3. Off-campus access to your VM.
4. Mirror the course git repository on your VM home directory.

[This document uses the term "workstation" to refer to the computer you are using to access your VM].

1.1 Initial Password Change

You should have received an email giving you the IP address `VM-IP` of your VM, your login id `VM-ID` and your initial password `VM-PW`. Since your password has been exposed in an unsecure email, it **must be changed**. Your should immediately login to your VM from the campus network and change your password to some `VM-NEW-PW` of your choosing.

The following log assumes that you are logging into your VM using `ssh` from `remote.cs`:

```
remote02:~$ ssh VM-ID@VM-IP
The authenticity of host 'VM-IP (VM-IP)' can't be established.
ECDSA key fingerprint is ...
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'VM-IP' (ECDSA) to the list of known hosts.
VM-ID@VM-IP's password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-33-generic x86_64)
...
VM-ID@COURSE-VM-ID:~$ passwd
Changing password for VM-ID.
(current) UNIX password: VM-PW
Enter new UNIX password: VM-NEW-PW
```

```
Retype new UNIX password: VM-NEW-PW
```

1.2 Administering Your VM

Your VM-ID account has administrative privileges. You can run any administrative CLI command by preceeding that command by `sudo` at which point you will be prompted for your current password. If you run an administrative task using the GUI, you will again be prompted for your current VM password.

Besides your account, you will find that the VM has 2 additional accounts. You should **not** touch these accounts. If something goes wrong with your VM, we may be able to use these accounts to get back in to your VM and attempt to fix the problem.

You can install additional system-wide software packages using the command line using apt-get or using a GUI-program found on the system menu.

The VM has text editors like `nano`, `emacs` and `vim` already installed. However, you may want to use the above package manager to install a text editor of your choice, or even an IDE.

1.3 Restarting your VM

In case you accidentally shutdown your VM or simply find it unreachable, you can use your VM's web console to restart it. Specifically, use the *vSphere web console* to access your VM's console; then use the `Actions->Power` menu item to restart your VM. Instructions for connecting to the VM's console are in the *vSphere Web Client section*.

1.4 GUI Access to your VM

If you use `ssh`, you basically obtain only command-line access to your VM. But the VM has the xfce4 desktop environment installed. Once you have GUI access to your VM, you can customize the GUI. Here are some of the alternative methods which you can use to obtain GUI access.

1.4.1 X2Go Access (Recommended)

1. Install a *X2Go client* for the workstation you are using to access your VM if one is not already installed (a `x2goclient` command should be available on `remote.cs`).
2. Start the `x2goclient`.

3. Create a session for connecting to your VM by clicking *Session -> New session*. Provide your VM's IP address `VM-IP` and your login id `VM-ID`. Specify the session type as XFCE.

4. You should now be able to use the above session to access your VM using a GUI.

Please note that if the campus sub-net you are using does not allow direct access to your VM, you will need to tunnel in to your VM. Please follow the instructions given below for x2go access using a tunnel.

1.4.2 VNC Access

VNC allows remote access to graphical desktops. It is not as efficient as x2go which is optimized for X11, but has the advantage that it can be used to access non-X11 desktops like a Windows desktop.

1. Connect to your VM using `ssh`.
2. Set up a ~/.vnc/xstartup script on your VM to start `xfce4`.
3. Start up a VNC server on your VM.
   (a) Run `vncpasswd` to set up a password `VNC-PW` for your VNC sessions.
   (b) Run `vncserver` to start up the VNC server. Note the VNC port `VNC-PORT` on which the server starts (usually 1).
4. Install a VNC viewer for the workstation you are using to access your VM if one is not already installed.
5. Run the VNC viewer to connect to `VM-IP:VNC-PORT`. Provide your VNC password `VNC-PW` when prompted. You should see your desktop.

1.4.3 vSphere Web Client

1. Go to this URL: *<https://csvb-vc.pods.bu.int/ui/>*
2. Login with your PODS username and password. Enter `PODS\` before your username
3. Once the vsphere-client loads, click on the `Menu` drop down and then click `VMs and Templates`
4. On the left-hand side of the interface you can expand the tree. Keep expanding the levels until you see a VM whose name is based on your PODS ID. You should see a summary of your VM.
5. Use the provided link to open a **web console** to the VM which will allow you to interact with it as if you were sitting in front of it using a console which runs within your web browser.

Note that unlike the x2go or vnc options, the vSphere web client will not persist your session on your VM.

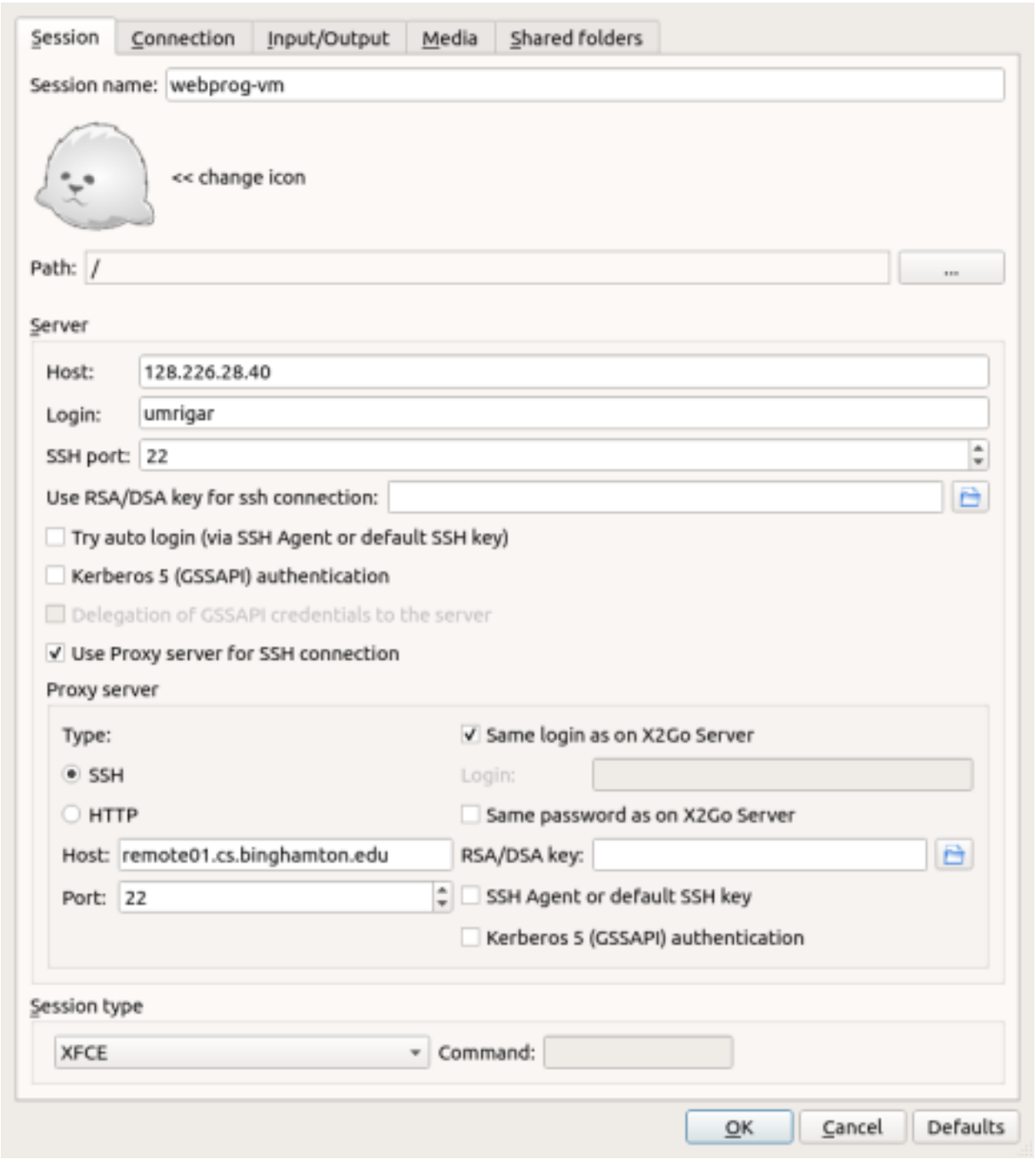
1.5 Off-Campus Access to your VM

You will need to tunnel in to your VM in order to access your VM from off campus. The exact method you use to establish the tunnel will depend on the method you use to access the VM and the OS used on your workstation. The following instructions assume the use of Unix-based (Linux, OS/X) workstations.

1.5.1 X2Go Access

When configuring your X2Go session, select 'Use Proxy server for SSH connection'. Provide the parameters for the proxy server: Type: SSH, your login on `remote.cs`; since `remote.cs` is a cluster, you may want to specify a particular `remote` for the host; say `remote01.cs.binghamton.edu`.

The following is a screen shot of my `x2go` configuration for accessing my VM from off-campus:



When connecting, I need to type in passwords for both my VM and `remote.cs`.

1.5.2 VNC Access

Assuming your VNC server is running on VNC port 1 (which corresponds to TCP port 5901), set up a tunnel via `remote.cs` using a script like the following:

```
SERVER=VM-IP:5901
LOCAL_PORT=5901
TUNNEL=YOUR-REMOTE-CS-LOGIN@remote.cs.binghamton.edu

ssh -fAX -L$LOCAL_PORT:$SERVER $TUNNEL -N
```

You should then be able to set up vnc access to your VM using a command like

```
vncviewer localhost:1 &
```

1.5.3 Command-Line Access

Set up a tunnel via `remote.cs` using a script like the following:

```
SERVER=VM-IP:22
LOCAL_PORT=2222
TUNNEL=YOUR-REMOTE-CS-LOGIN@remote.cs.binghamton.edu

ssh -fAX -L$LOCAL_PORT:$SERVER $TUNNEL -N
```

You should then be able to set up ssh access to your VM using a command like

```
ssh -p 2222 VM-ID@localhost
```

1.6 Mirror Course Git Repository on your VM

1. On your VM, create a public-private key-pair:

   ```
   $ ssh-keygen
   ```

   If you accept the defaults, you will create a private key in `~/.ssh/id_rsa` and the corresponding public key in `~/.ssh/id_rsa.pub`. The former should be retained on your VM whereas the latter may be distributed to machines to which you would like to access from your VM.

2. Copy the public member of the above key-pair to `remote.cs`:

   `$ ssh-copy-id -i ~/.ssh/id_rsa` *LOGIN*`@remote.cs.binghamton.edu`

where *LOGIN* is your login-id on `remote.cs`.

You will be prompted for your password on `remote.cs`.

3. Create a ~/git-repos top-level directory to hold all your `git` projects and go into it.

```
$ mkdir ~/git-repos
$ cd ~/git-repos
```

4. Clone the course web site:

`$ git clone` *LOGIN*`@remote.cs.binghamton.edu:~umrigar/git-repos/cs580w.git`

where *LOGIN* is your login-id on `remote.cs`. All the files contained within the course web site should be copied into a `cs580w` sub-directory.

5. Go into the `cs580w` directory. You should then see all the course web site files.

6. Create a symlink to the `cs580w` directory in your home directory:

```
$ cd
$ ln -s git-repos/cs580w .
```

This will ensure that scripts can find the directory.

You should never be writing into this directory. Periodically update this directory:

```
$ cd ~/git-repos/cs580w
$ git pull
```

Use `git log` to see a brief description of the commits made since you last updated your directory.

You should treat this as a read-only repository; you can always update your mirror by going into `~/cs580w` and typing `git pull`. You can also consider setting up your crontab to periodically `pull` it.

1.7 Manually Tracking cs580w Changes

You can get a summary of all git changes by running a Ruby script:

`$ ~/cs580w/bin/git-changes.rb`

This will output a summary of all git changes since the last time you ran the script (it records the time via the timestamp on `~/cs580w/.last-login`).

If you want to see all the commits in a particular directory like `hws/hw1`:

```
$ cd ~/cs580w
$ git log --oneline -- hws/hw1
dcdef92 minor hw1 changes
05fc1a1 added hw1; minor correction to prj1
$
```

Your commit id's may differ.

If you want to see the details of a particular commit, use:

```
$ git log --stat dcdef92^!
```

To see the diff's for a particular commit-id:

```
$ git show dcdef92
```

To restrict the diff's to a particular path:

```
$ git show dcdef92 -- hws/hw1/hw1.umt
```