BLCN532 Lab 1
Set up your development environment

Introduction

This course introduces students to blockchain development for enterprise environments. Before you can develop software applications, you need to ensue your development environment is in place. That means you'll need all the tools and infrastructure installed and configured to support enterprise blockchain software development projects.

In this lab you'll set up your own Hyperledger Fabric development environment and install the course software from the textbook. When you finish this lab, you'll have a working development environment and will be ready to start running and modifying blockchain applications.

The instructions in your textbook are for Mac and Linux computers. If your computer runs Microsoft Windows, just follow the instructions in the section "Initial setup for Windows computers" first.

Lab Deliverables:

To complete this lab, you must create a **Lab Report file** and submit the file in iLearn. The Lab Report file must be a Microsoft Word format (.docx), and have the filename with the following format:

BLCN532_SECTION_STUDENTID_LASTNAME_FIRSTNAME_Lab01.docx

- SECTION is the section number of your current course (2 digits)
- STUDENTID is your student ID number (with leading zeros)
- LASTNAME is your last name
- FIRSTNAME is your first name

To get started, create a Microsoft Word document (.docx) with the correct filename for this lab. You'll be asked to enter text and paste screenshots into the lab report file.

NOTE: All screenshots MUST be readable. A screenshot that I cannot read (i.e. cannot read the text you are capturing) will not count for any points.

SECTION 1: Initial setup for Windows computers (Chapter 3)

Step 1.1: Install Oracle Virtualbox

Oracle Virtualbox is an open source virtualization environment that allows you to run multiple virtual machines and containers on a single personal computer. Virtualbox is free and it is easy to install.

In your favorite web browser, navigate to: https://www.virtualbox.org/ and click the "Download Virtualbox" button. Click the "Windows hosts" link to download the main installation executable. You should also click the "All supported platforms" under the "Extension Pack" heading to download extra software support for devices.

After you download the two files, double click each one to run the install procedure.

Step 1.2: Install Vagrant

Vagrant is a free virtual environment management utility. It makes the process of starting, stopping, and managing virtual machines easier. In your web browser, navigate to https://www.vagrantup.com/ then click the "Download" button, and click the version of the Windows executable you'd like to install. (Most of you should select the "64-bit" version.)

Once you download the install program, double-click the file you just downloaded to install Vagrant.

If you want more information on Vagrant and tips on getting the most out of the software, navigate to:

https://www.sitepoint.com/getting-started-vagrant-windows/ .

Step 1.3: Install PuTTY

Later in this lab you'll launch several Linux virtual machines. These VMs won't have a graphical user interface, so you'll need a way to log in and use the VM's command line. You'll use a terminal emulator program, called PuTTY to do that. PuTTY is a free and is one of the most popular terminal emulator programs.

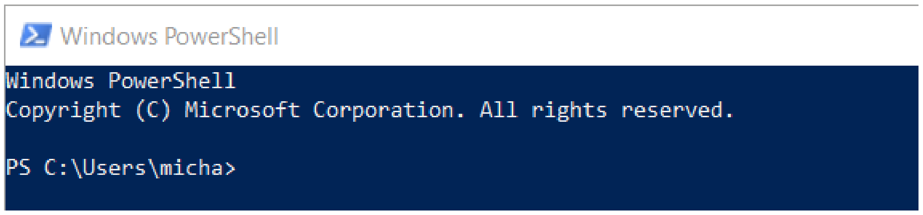
To download and install PuTTY, navigate to https://www.chiark.greenend.org.uk/~sgtatham/putty/, then download and install the PuTTY program.

Step 1.4: Set up your Vagrant project

After installing all the pre-requisite pieces, you need to set up your Vagrant project. A Vagrant project defines your virtual machine environment and helps you organize your collection of VMs into a group that is easy to manage.

We'll use the Windows PowerShell as our Windows command prompt environment. PowerShell is a very powerful command line interface that is available on all Windows computers.

To launch PowerShell, click the Windows key, type PowerShell, then click the Windows PowerShell menu entry. The figure below shows a portion of the Windows PowerShell command prompt window.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\micha>
```

Note that PowerShell uses your user's home directory as its starting directory. In my case, C:\Users\micha is my home directory. For the rest of the lab, I'll refer to this a %HOME%. Your %HOME% directory will be different.

1.4.1: Remove existing Vagrant projects

Follow these steps ONLY if you already have a previous Vagrant project you want to remove: (Assume the project you want to remove is located in the %HOME%\vagrant\Hyperledger directory.)

If you DO NOT have an existing Vagrant project that you need to remove, skip to the next paragraph.

1. PS %HOME%\vagrant\Hyperledger> **vagrant global-status**

   Note the id of the listed VM(s). You'll use this id in the next command, in place of xxxxxxx.

2. PS %HOME%\vagrant\\Hyperledger> **vagrant destroy xxxxxxx**
3. PS %HOME%\vagrant\\Hyperledger> **vagrant box remove ubuntu/xenial64**

1.4.2: Create a new Vagrant project for Hyperledger

Launch PowerShell and enter the following commands: (Don't type 'PS %HOME%>', that's just the PowerShell prompt. Just type the characters in bold.)

1. PS %HOME%> **mkdir vagrant**
2. PS %HOME%> **cd vagrant**
3. PS %HOME%\vagrant> **mkdir Hyperledger**
4. PS %HOME%\vagrant> **cd Hyperledger**
5. PS %HOME%\vagrant\Hyperledger> **vagrant init ubuntu/xenial64**

6. PS %HOME%\vagrant\Hyperledger> **vagrant up**

NOTE: To stop your VM, exit from PuTTY shell, then type vagrant halt in PowerShell

1.4.3: Set up PuTTY SSH connection

Now that you have a working VM, you need to set up PuTTY to be able to easily log into your VM. That's how you'll access your VM's command line to enter commands for the rest of the class. In this lab step you'll set up PuTTY to allow you to login using a private key, instead of having to enter a userid and password every time you connect. You'll use a PuTTY utility, PuTTYGen, to generate your public/private key pair, and then load your key into PuTTY's configuration.

1. Ensure that your VM is started (PS %HOME%\vagrant\Hyperledger> **vagrant up**)
2. PS %HOME%\vagrant\Hyperledger> **vagrant ssh-config**
3. Launch PuTTYGen (type **puttygen** at the PowerShell prompt)
4. Click **Load,** then navigate to %HOME%\vagrant\hyperledger\.vagrant\machines\default\virtualbox\private_key

Make sure to select All Files (*.*) to the right of the filename input

5. Select **OK -> Save Private Key -> Yes -> private_key_putty -> Exit**
6. Launch PuTTY
7. Type the following information, then click **Save**:

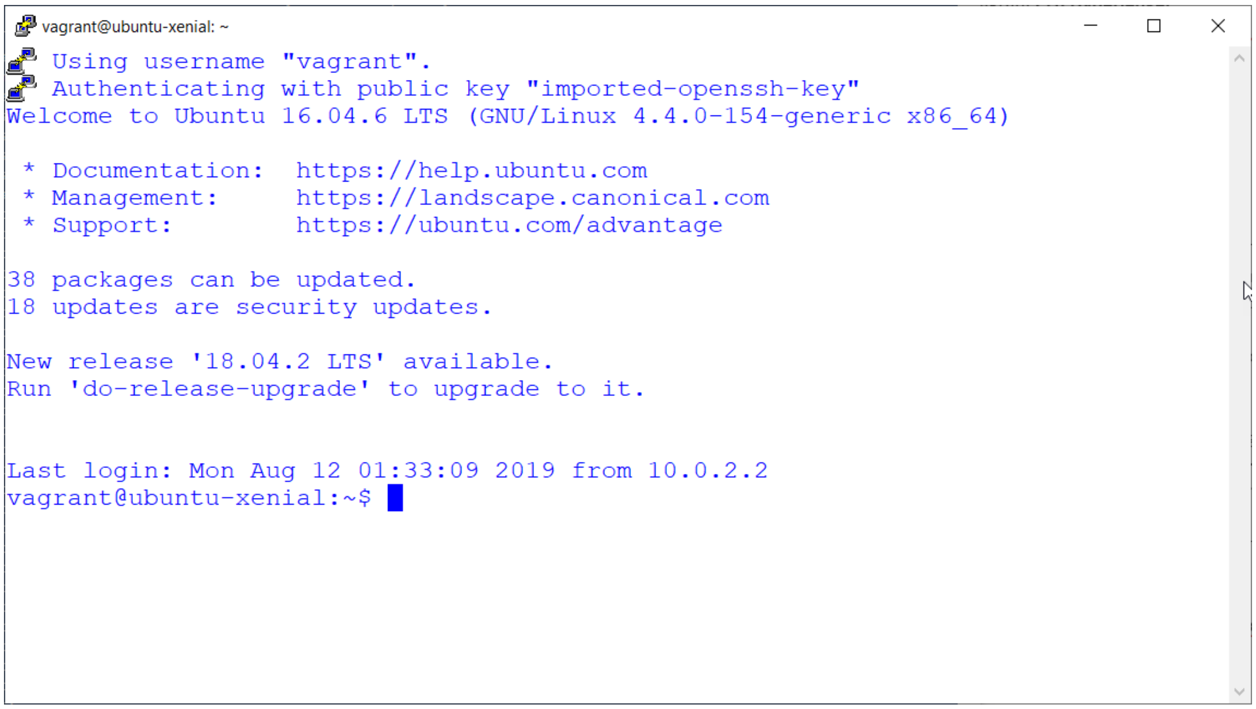
   Host Name (or IP address): **127.0.0.1**

   Port: **2222**

   Saved Sessions: **Vagrant Hyperledger**

8. Click Connection -> SSH -> Auth -> Browse -> %HOME%\vagrant\hyperledger\.vagrant\machines\default\virtualbox\private_key_putty
9. Click Connection -> Data -> Auto-login username: **vagrant**
10. Open connection

You should see a command prompt like this: (I've changed my font and colors, so yours will look appear different, but the contents should be the same.)


vagrant@ubuntu-xenial: ~

```
  Using username "vagrant".
  Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-154-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

38 packages can be updated.
18 updates are security updates.

New release '18.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Mon Aug 12 01:33:09 2019 from 10.0.2.2
vagrant@ubuntu-xenial:~$
```

Section 2: Install Pre-requisites (Chapter 3)

Once you have an operating Linux VM, you can start to install the Hyperledger Fabric pre-requisites. In this section you'll install all the software and configuration pieces necessary to run the class enterprise blockchain application in Hyperledger Fabric.

Step 2.1: Install pre-reqs

2.1.1: Install dev tools

You type all the following commands in your Linux VM (at the command prompt using PuTTY.)

1. $ **sudo apt-get update**
2. $ **sudo apt-get install make**
3. $ **sudo apt-get install libltdl-dev**

2.1.2: Install Docker-CE

You can find complete Docker-CE instructions/docs at: https://docs.docker.com/install/linux/docker-ce/ubuntu/

1. $ **sudo apt-get remove docker docker-engine docker.io containerd runc**
2. $ **sudo apt-get install \**

   apt-transport-https \

   ca-certificates \

   curl \

   gnupg-agent \

   software-properties-common

3. $ **curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -**
4. $ **sudo add-apt-repository \**

   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \

   $(lsb_release -cs) \

   stable"

5. $ **sudo apt-get update**
6. $ **sudo apt-get install docker-ce docker-ce-cli containerd.io**

2.1.3: Install Docker compose

1. $ **sudo curl -L "https://github.com/docker/compose/releases/download/1.23.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose**
2. $ **sudo chmod +x /usr/local/bin/docker-compose**
3. $ **docker-compose –version**
4. Create a screenshot of the results of steps 1, 2, and 3, and paste that screenshot into your Lab Report File.

2.1.4: Install business network pre-reqs

1. $ **curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh**
2. $ **chmod u+x prereqs-ubuntu.sh**
3. $ **./prereqs-ubuntu.sh**
4. Create a screenshot of the results of steps 1, 2, and 3, and paste that screenshot into your Lab Report File.

>>> Logout and log back in after completing the previous step

2.1.5: Install GO language

The next step is to install the GO programming language. You'll use GO to write and modify source code files for the class blockchain application.

1. $ **sudo apt-get update**
2. $ **wget https://dl.google.com/go/go1.12.1.linux-amd64.tar.gz**
3. $ **tar xvf go1.12.1.linux-amd64.tar.gz**
4. $ **nano ~/.profile**
5. In the nano editor, add the following 2 lines after the last line in the file:

   export GOPATH=$HOME/go

   export PATH=$PATH:$GOPATH/bin

6. Save your file and exit nano
7. $ **source .profile**
8. $ **go version**
9. Create a screenshot of the results of steps 1-8, and paste that screenshot into your Lab Report File.

Step 2.2: Install source code and set up business network

In this step you'll download and install the class application source code and set up the class business network.

2.2.1: Forking and Cloning the trade-finance-logistics repository

1. In a browser (**in Windows**) go to: https://github.com/HyperledgerHandsOn/trade-finance-logistics
2. Sign up or log in
3. Click **Fork**

You type all the following commands in your Linux VM (at the command prompt using PuTTY.)

4. $ cd $GOPATH/src
5. $ git clone https://github.com/YOUR_GIT_ID/trade-finance-logistics.git
6. $ cd $GOPATH/src
7. $ mkdir -p github.com/hyperledger
8. $ cd github.com/hyperledger
9. $ git clone https://github.com/hyperledger/fabric.git -b release-1.1
10. $ cd fabric
11. $ make docker
12. $ make configtxgen cryptogen
13. $ git clone https://github.com/hyperledger/fabric-ca.git
14. $ cd fabric-ca
15. $ make docker
16. Create a screenshot of the results of steps 1-15, and paste that screenshot into your Lab Report File.

2.2.2: Generate network cryptographic material

1. $ cd $GOPATH/src/trade-finance-logistics/network
2. $ nano ~/.profile

   change last line to this (all on 1 line, not 2):

   export PATH=$PATH:$GOPATH/bin:$GOPATH/src/github.com/hyperledger/fabric/build/bin:$GOPATH/src/github.com/hyperledger/fabric/build/docker/bin

3. Save file and exit nano

4. $ source ~/.profile
5. $ cryptogen generate --config=./crypto-config.yaml
6. Create a screenshot of the results of steps 1-5, and paste that screenshot into your Lab Report File.

2.2.3: Generate channel artifacts

1. $ cd $GOPATH/src/trade-finance-logistics/network
2. Execute the follow commands OR run the trade.sh script
3. $ mkdir -p channel-artifacts
4. $ configtxgen -profile FourOrgsTradeOrdererGenesis -outputBlock ./channel-artifacts/genesis.block
5. $ configtxgen -profile FourOrgsTradeChannel -outputCreateChannelTx ./channel-artifacts/channel.tx -channelID tradechannel
6. $ configtxgen -profile FourOrgsTradeChannel -outputAnchorPeersUpdate ./channel-artifacts/ExporterOrgMSPanchors.tx -channelID tradechannel -asOrg ExporterOrgMSP
7. $ configtxgen -profile FourOrgsTradeChannel -outputAnchorPeersUpdate ./channel-artifacts/ImporterOrgMSPanchors.tx -channelID tradechannel -asOrg ImporterOrgMSP
8. $ configtxgen -profile FourOrgsTradeChannel -outputAnchorPeersUpdate ./channel-artifacts/CarrierOrgMSPanchors.tx -channelID tradechannel -asOrg CarrierOrgMSP
9. $ configtxgen -profile FourOrgsTradeChannel -outputAnchorPeersUpdate ./channel-artifacts/RegulatorOrgMSPanchors.tx -channelID tradechannel -asOrg RegulatorOrgMSP

OR (Instead of typing the commands above, the author provides a script to do it all)

1. $ ./trade.sh generate -c tradechannel

2.2.4: Launch the sample trade network

There are two ways to launch the sample trade network, using the **docker-compose** command directly, or by using the author-supplied **trade.sh** shell script. We'll use the shell script in most of our activities.

1. $ **cd $GOPATH/src/trade-finance-logistics/network**
2. $ **./trade.sh up**
3. Create a screenshot of the results of steps 1 and 2, and paste that screenshot into your Lab Report File.

Step 2.3: Verifying your business network

The last step in this lab is to verify that your business network is up and running.

1. Open a new PuTTY session for your Vagrant Hyperledger VM. (**Right-click** on the PuTTY icon in the task bar and **click** PuTTY.)
2. Enter the following command:
3. $ **docker ps -a**
4. Create a screenshot of the results of step 3 and paste that screenshot into your Lab Report File.
5. The first column in the previous output is the "container-ID". Find the container-ID for the orderer process, and use it for the following command: (Don't type the "<" or ">" characters)
6. $ **docker logs <container-ID>**
7. Create a screenshot of the results of step 6 and paste that screenshot into your Lab Report File.

Section 3: Wrapping up

Once you have your network up and running, the only step to complete is to learn how to properly shutdown your business network.

1. Shut down the business network in Linux:
2. $ **./trade.sh down**
3. Exit from your Linux session
4. $ **exit**
5. In Windows PowerShell, shut down your Hyperledger virtual machine:

7. PS %HOME%\vagrant\Hyperledger> **vagrant halt**

You should have 8 screenshots in your Lab Report File. Save your file and submit it in iLearn as a file attachment for the Lab 1 assignment.

Congratulations! You have complete lab 1.