BRG27 –   
ISEA Introduction to Server Environments and Architecture

Reflective learning Journal

Name: Tee Wei Aun, Ken

Student ID: CT0380834  
  
Github link : [KenTee96/BRG-27-labs: Documentation for ISEA bridging module](https://github.com/KenTee96/BRG-27-labs)

# Day 1 - Setting up and Exploring Linux

For the first day, I first set up and updated a GitHub profile, creating a repository that will hold an overview of the activities performed over the course of the ISEA lab and classes. I started off with a README.md that will be used as the overview in the main repository branch. The repository also contains screenshots of steps taken and on certain activities that was performed throughout the duration of the course.

Thereafter, I proceeded to download VirtualBox. I had to learn how to differentiate all the different types of Linux, and the commonly used ones like Ubuntu also have a wide variety of different flavors. As for the setup and configuration, the different .iso files would also be a challenge, as the commonly used distros available downloads of different versions, there are lite versions, long term support versions etc. A read up have to be done to know which .iso I would like to use for configuring the virtual machine.

I decided on two different distributions of Linux to test, try and have and comparison for: firstly, the latest version of Ubuntu, secondly, Kali Linux (mainly as I am working in a Cybersecurity sector and would like to be familiarized with this commonly used distribution).

Loading up Virtualbox, I then proceeded to add new Virtual machines. The initial configuration would include setting of the username, allocation of resources for the virtual machine like the RAM and the disk space to be set for it, and the .iso file to be used for booting the virtual machine before the installation of the OS. The installation was quite easy as these commonly used hypervisor now have guided installation processes where the input of certain things like user and administrative setup can be input prior to the OS installation.

After the virtual machines have been set up, I went onto terminal on Ubuntu and Kali. The interfaces look very slightly different. I tried common commands like pwd, ls, man, mkdir, rm, mv, touch, cat, and a little on the editor nano. Initially had used ‘man’ to read more into the commands, learnt a little on the arguments that can be added.

**(Challenges)** I faced my first real challenge of the activities: I have allocated too little RAM during the configuration. When trying to run update packages and having multiple simple services running later on, I noticed that processes get killed automatically and was wondering why, and reading up, the most common would be OOM (out of memory) issues. I then had to re-configure the allocation of RAM and repeat the process of setting up the VM, giving extra bandwidth so that this issue can be avoided as a whole.

For Github, I do not know how to create or place my screenshot uploads into a separate folder, and instead, just simply commited the uploads to the main branch, therefore making it a little messy as all the activities that I have placed into screenshots are just on the main branch without any proper indexing. Looked into the creation of subfolders and had to clone the image uploads into those subfolders for more organization.

# Day 2 - Sevices, Cost and Scripting

For Day 2, I started working on looking at the various service providers for cloud networks. Two of the main considerations were Azure and AWS resepectively. Signed up for both and with my student email, managed to accrue free credits for the Microsoft Azure platform and ultimately chosen Azure as I am also dealing with it for work. AWS offered similar services and seem to be also quite streamlined and guided, but due to my potential need to face the Azure environment, proceeded with Azure.

I then had to choose the Azure VM type, and the costings were clearly indicated, I took note of some of the details in terms of costs so that I can practice it for the Total cost of operation practice. For the TCO practice, I did a very simple excel sheet, documentating the use of VM and the estimated cost after the setup, more details towards the end of this section.

I proceed with the setup and chosen Size **“Standard B2ats v2”** as it was at a subsidized cost for the student account (did not read too much into that). I then went onto the account cost estimate and took a screenshot, the estimated cost for the month of July during this course would be $4.20 USD, approximately $5.30 SGD at the time of writing. Used this data to compare. The 3 and 5 years cost projection assuming this is the only VM would be $151 and $252 respectively. I then took a look at the secondhand market for used laptops and it was displayed to have cheapest ones at around $300 USD for older laptops that have the minimum specifications to host the virtual machines with my own hardware if needed, that would be without the costing of the power usage or internet at home.

For my own application in this ISEA course of studies, Cloud VM seemed to be the feasible option in terms of costing in this case.

Back to the technical aspect for Day2, after the setting up and cost analysis of the virtual machine, I then proceeded to do some simple OS updates, similarly, tried some simple Linux commands, and created a directory for Scripts.

I tested a simple script and had some issues as I did not manage to get the syntax right the first few times around, and had to use nano the editor to edit it accordingly while learning the syntax. I then did up a script for disk usage reports that echo the data of disk usage to a .txt file, and an update.sh where it runs the OS update.

Thereafter, moving on the the last portion of the day, I managed to mody the permissions for the script files. Ccrontab -e was also used to edit and create scheduled tasks for some of the scripts I have made to run. I have set the update script to run at night before the Virtual machine’s auto shutdown.

# Day 3 - DNS, Certificates, and Automation

Day 3 felt a bit more technical, but it seemed to be more straightforward as I started working on actual public access configuration for the Azure VM that was created for day 2 activityes. The main objective I had for the day was to host a small simple webpage and get it linked through a domain name rather than just using the public IP address given by Azure. To do this, I explored the free dynamic DNS services online and eventually went with DuckDNS, as it offers a simple and cost-free way to assign a subdomain to a public IP. The other services had more freedom to the main domain, and even getting vraitions like for example, Ken.ai, ken.lab etc.

I registered for a free DuckDNS subdomain with the name iseacloudlab.duckdns.org. There were some steps involved such as creating a token from DuckDNS and having to run a simple curl-based update command to keep the IP updated. I did some reading and realized the IP can be auto updated with a cronjob later, butthere was no need as the IP have been set to be static. Once the DNS part was set up, I tested it using dig, just to confirm the domain name is resolving to my VM correctly, and the result was that the information to the domain name linked back to my VM’s public IP.

Configuring a web server., I installed Nginx using sudo apt install nginx and checked that the default landing page was working via my new subdomain.However, accessing the web on my browser brought it to a http site (which will be the reason for the last part of the lab involving certbot). After confirming Nginx worked, I went ahead to modify the default index page. I created a very simple index.html page with a short greeting message, just a line or two to confirm the server’s running with content. This took some time as it was a long time ago when I worked with html. I also had certain difficulties determining the directory that the index.html was located (had not configured nginx myself before)

On top of that, I also added a text file called testdoc.txt to test out static file serving. Linked it as a hyperlink in the the index.html and then accessed it via the browser. I wanted to do this as I would like to also see the access of my page, the request received by my server and also to possible implement some sort of security features that help to block off IP that are detected to send request with potentially malicious intent.

For the last part of the day, I went on to set up the secure connection for HTTPS. I went on to install Certbot for SSL certification. This was also my first time doing this, and I ran into a small issue with Certbot’s missing initially, but managed to sort it out by reading through the official documentation. After completing the Certbot process, HTTPS was successfully enabled for my subdomain. Accessing the site through http and https ports loaded the page in https. This was also quite straightforward as the default configgurations on certbot itself would do things like force http connection to the https connectiuon.

At this stage, the server had its own domain name, was serving my custom index.html page, logging access activity (throught the default access.log from nginx), and even had HTTPS implemented. It made me realize how accessible server deployment has become, and how just a few configurations can bring a VM to life on the public internet.

# Day 4 - Consulting Simulation and Final Reflection

For the day 4, I decided to explore something I’ve never worked with before—setting up an **FTP file server**. I chose to use **vsftpd** since it’s widely recommended for Ubuntu and known for its security. Installation was simple with sudo apt install vsftpd, but configuration was where things got a bit tricky.

At first, I couldn't connect at all through FileZilla—kept getting errors like “login failed” or “connection refused.” After digging through online guides and the /etc/vsftpd.conf file, I realized some settings like local\_enable=YES and write\_enable=YES were required to allow user logins and file uploads. Once I enabled those and restarted the service, I made progress.

I also created a ftpuser and set some permission to have no write permissions when. Solved that by adjusting ownership and permissions using chmod and chown. Once everything clicked, I could download files through my local machine’s powershell.

On the security side, I enabled **UFW** and allowed FTP ports 20 and 21 alongside SSH. I ran sudo ufw allow 20/tcp and 21/tcp, and confirmed it with ufw status. It was simple, but effective for restricting traffic.

As a small side task, I revisited the **access log** idea from Day 3 and used grep to identify IPs making strange requests (or many requests in a short period of time where it is not needed for a simple webpage like mine), and so on. Didn’t automate any blocking yet, but noted a few IPs and saved the command for future use.

Despite the initial confusion with FTP settings, I learnt a lot about how file servers work and how to manage access securely. It was a fitting final task—bringing together networking, permissions, and security into one practical setup.

Overall Challenges and reflection sumaryOver the course of four days, I have learnt a lot about server architectures and environments, trying out powershell, bash terminal, linux OS (Kali and Ubuntu), did a lot of practice for linux commands, getting to the know the linux environment and it’s structure, scripted, hardened the server with permission changes, even hosted a site with a domain registered.

Most of the challenges I faced comes with the wide variety of available service providers, and the number of tools and services available too would be overwhelming. Other than that, Syntax and familiarity with the type of system would also be one of my main challenge faced, but that can be easily overcome with reading, and practice (man command and online resources are in great detail).