jason.pang@senecacollege.ca

Please use a Screen or Video Capture software to save your works!

OBJECTIVE & PREPARATION

In this lab, we will look at several separate technologies that are used with the Apache web server to install, configure and run web applications.

The basic purpose of the Apache web server is to serve text pages, images, and other static files. You can format those webpages to appear nice, but they would lack dynamic functionality (i.e., the ability to change colors or font-size when the mouse moves over a link, button, etc.). In other words, using Apache webserver as a sole application would not make your webpages more interesting.

If we want to add more features for our webpage (e.g. dynamic functionality, security, e-commerce, etc.), your webserver would need additional help. To provide additional help requires several resources - more than just the web server itself. A popular acronym to represent these foundations and servers is referred to as **LAMP**. It stands for **Linux, Apache, MySQL**, and **PHP** (or Python).

In your previous OPS245 course, your second assignment may have required you to setup a similar series of services in order to run a Wiki on one of your virtual machines. In this lab, we will set up another example of a "**LAMP solution**" that will allow the user to run webmail in a web-browser to send and receive e-mail messages.

Online Resources

Installing Apache Webserver on Centos7

PHP Tutorial (w3schools.com)

MySQL / SQL Language Resources (w3schools.com)

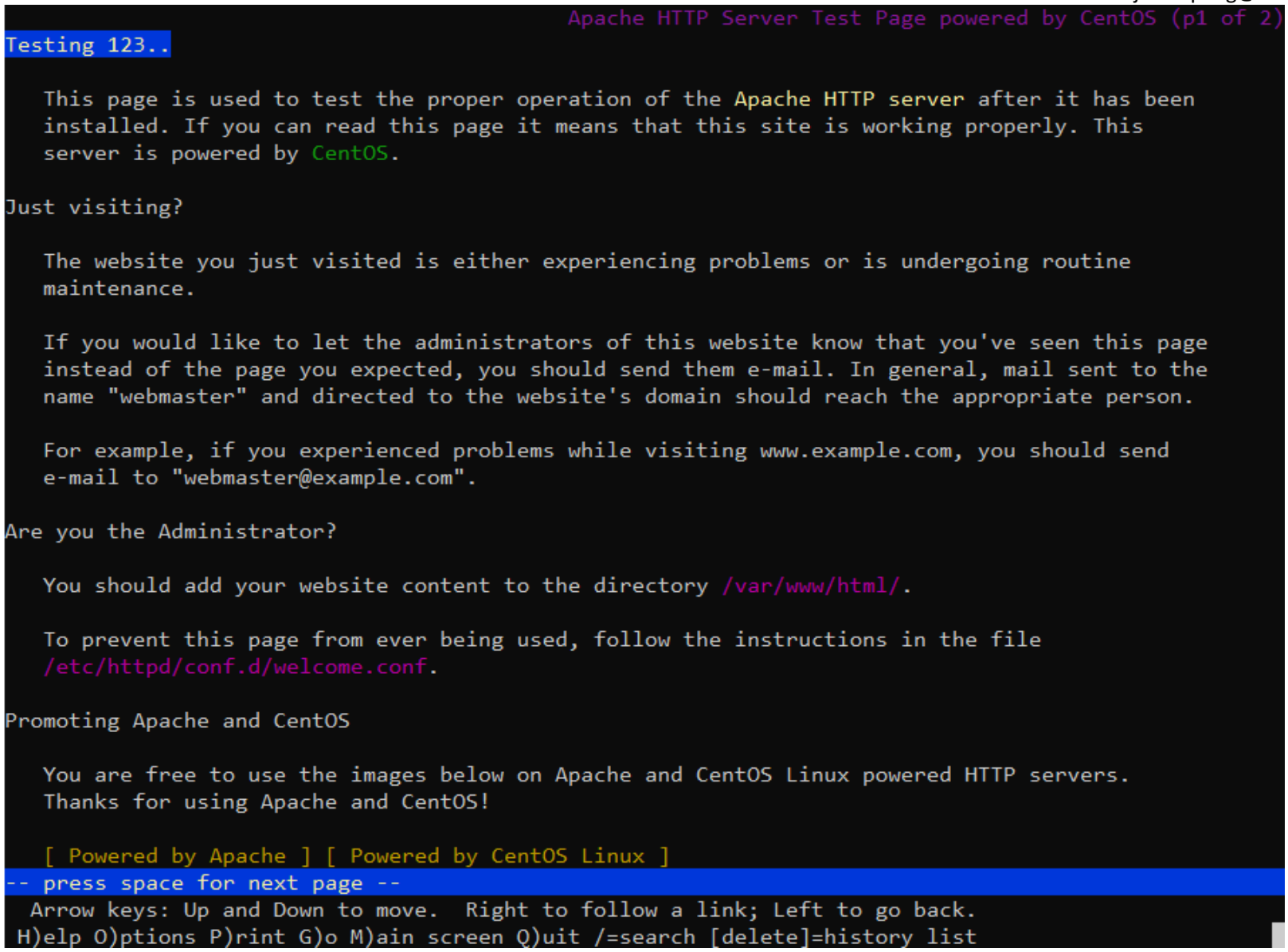
Case 1: Setting Up a Webserver With Dynamic Webpages

Install, Configure & Run a Webserver (Apache)

We need to install, configure and run a webserver on one of our Linux VMs.

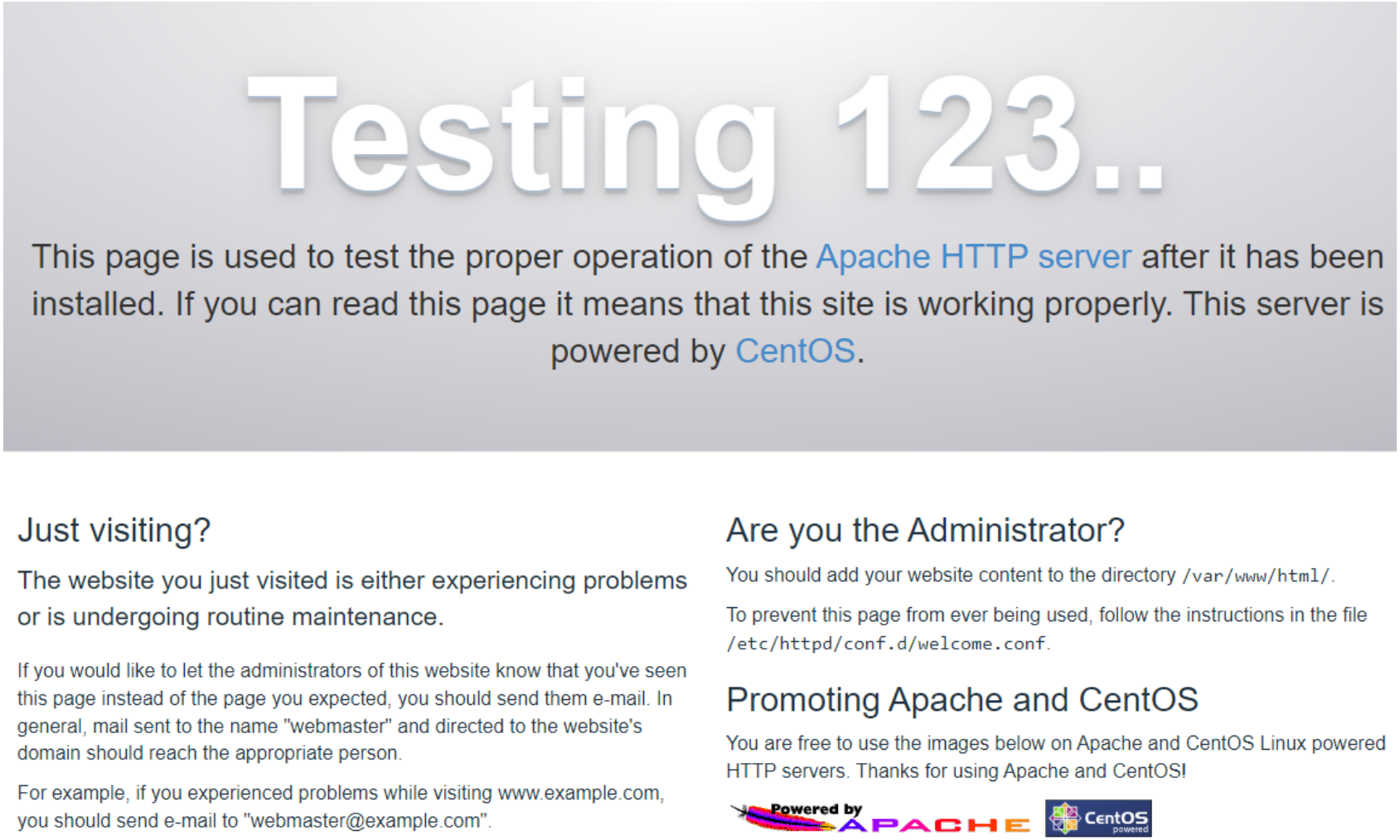
Perform the following steps:

1. Make sure you are in your **Server** VM.
2. Install the Apache package (the name of the package is **httpd**).
3. **Start** the **httpd** service and **enable** this service to start automatically upon system start-up.
4. Using a text browser such as **lynx** on Server VM go to http://localhost You should get the Apache Test Page which indicates your web server is running on the local virtual machine.
    a. If **lynx** not installed yet. Install the **lynx** package.



5. Make sure to configure your firewall to allow access to the **httpd** service (i.e. the Apache serves HTTP traffic which goes over **TCP port 80** and **keep/save** the iptables changes past rebooting.
   a. HTTPS run through port 443, but we don't use https:// in this course.

6. Open a web-browser in your Windows VM and enter the following URL: http://server.youruserid.ops. (or the server's IP address if the DNS server not configured).
   a. Create new Windows VM if not yet created already, choose 2 vCores and 4GB memory for performance. This VM will only stay for this lab, to be removed at the end. (Keeping a Windows VM in Azure is rather expensive to your free credit. We will create the Windows VM **on demand**.)
   b. If you have setup your Apache webserver correctly, you should be able to view the Apache Test page.



7. Although we will not be exploring webservers in depth, we will have you create a simple webpage for testing purposes, then later setup a web resource for webmail.
8. The term **DocumentRoot** (in /etc/httpd/conf/httpd.conf) specifies where the Apache webserver will search for documents to serve. Create the file **index.html** in your **DocumentRoot** directory with the following contents (replace the date with the current one):
   Hello, this is a web page on server.youruserid.ops and the current time is Nov. 15, 22 15:00:00 EDT 2022!

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/var/www/html"
```

9. If you refresh your web page in your browser, you should see the contents of your index.html document. If you wish, you can specify the filename index.html in the address, but it is not necessary, since the file index.html is automatically loaded by default when the URL refers to that directory containing that file.

   Hello, this is a web page on server.jasonpang.ops and the current time is Nov. 15, 22 15:00:00 EDT 2022!

10. Refresh your webpage by issuing the key combination: **ctrl-r**. Notice that the time doesn't change as you refresh the page. This indicates that the page is static (not dynamic), indicating that the page does not change (i.e. boring!).
11. Create a folder called private under "/var/www/html" and then create a file and a folder inside the newly created private folder "/var/www/html/private"
    a. Now navigate to the http://youruserid.ops/private what do you see?

Using the index.html file

It is considered to be a "best practice" to create **index.html** files for newly-created subdirectories within the **DocumentRoot** (or users' public_html directories) to force a display of a web-page, instead of viewing the directories "index" listing of files (from "curious eyes"): that is why the name of the file is called "index.html". you can disable the directory browsing with the following option.

```
# Further relax access to the default document root:
<Directory "/var/www/html">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
    #
    # The Options directive is both complicated and important.  Please see
    # http://httpd.apache.org/docs/2.4/mod/core.html#options
    # for more information.
    #
    Options -Indexes
```

Creating a PHP Script

In order to allow us to run a webserver application in a web-browser, we need a scripting language that will allow the web-browser to function dynamically (i.e. being able to change frequently, as opposed to being "static" or unchanging). In this section, we will demonstrate how a scripting language (PHP) can be used for the web-browser to react in a more dynamic fashion.

PHP Scripting Language

PHP code is considered to be a language that runs on the web-server (i.e. "server-side programming"). PHP code can be embedded in an HTML document (HTML code), and use the resources on the "server-side" to make the web document or resource more dynamic (eg. database access, etc). Although it is not the purpose of this course to learn about and create PHP documents, here is a quick resource on PHP: PHP Tutorial

Perform the following steps:

1. **Copy** the **index.html** file as **index.php** and modify it to contain:
   Hello, this is a web page on server.youruserid.ops and the current time is <?php system("date"); ?>!
2. On your Windows VM, in the web-browser manually add **/index.php**. Notice that in a web browser the index.php file isn't treated as a default page and the contents don't contain the date, but instead are displaying the text in the php code you entered into the index.php file.

```
Hello, this is a web page on server.jasonpang.ops and the current time is <?php system("date"); ?>!
```

3. The reason this occurs is that the PHP interpreter hasn't been installed on your server by default.
4. Install the **php** package on your Server VM and restart your web service (**httpd**). NOTE: The php package comes with a working default Apache configuration so you don't need to enable or modify it manually.
5. Refresh the webpage in your web browser on your Windows VM. You should now notice that you see the date instead of the call to the date command. Refresh your webpage several times to see how the time

changes. This is simply a "trivial example" of dynamic web content provide a simple demonstration of how scripting languages can be used to create more dynamic webpages.

Hello, this is a web page on server.jasonpang.ops and the current time is Sat Nov 19 12:58:56 EST 2022 !

Controlling Access to Pages

For security, it is important to allow access to general areas of your webpage, but also limit access to other sub-directories that contain other webpages or documents. Penetration Testers or hackers may be able to navigate your file systems within your html directory to obtain unauthorized information.

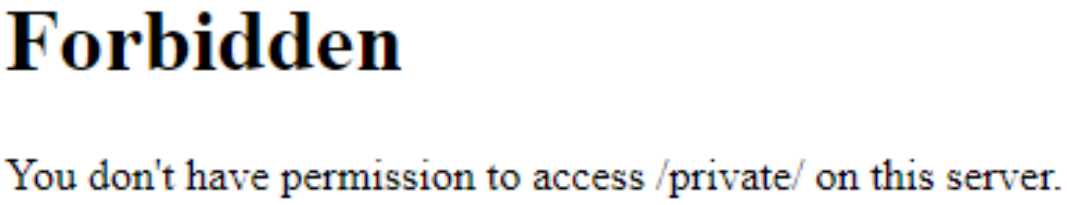
There are many common-sense safeguards, such as creating an index.html file in your default directory that will display a webpage instead of the directory index. On the other hand, there are also safeguards that you can setup to provide additional protection to your data on your web server.

Perform the following steps:

1. Move **index.php** to the newly created private directory.
2. View both index.html and index.php files.
   You will now modify the settings on the webserver to prevent machines outside our network from accessing the private directory.
3. Add the following directory statement to your apache configuration file. The default pathname for the apache configuration file is: /etc/httpd/conf/httpd.conf (NOTE: replace the X with your own network octet):
   Do not overwrite existing settings
   There should already be two Directory statements in that file. One for /var/www and one for /var/www/html. Add your new Directory statement after them. Do not overwrite them.

```
<Directory "/var/www/html/private">
        AllowOverride None
        Require ip X.X.X.0/24
</Directory>
```

4. This sets up separate rules and access permissions for that subdirectory. You should no longer be able to access any pages in the private directory (or any sub-directories of it) from external machines (connect to your server vm with IP from your home computers), but your internal machines (including your windows vm on Azure) should still have access.



Case 2: Setting Up an Online Database

The next piece of the puzzle is installing, configuring, and running a database server to support your webmail application that will be installed and setup later in this lab.

Install, Configure and Run MySQL Database Server

MySQL is used to allow storage and retrieval of structured data. SQL is a command language (used by scripting languages such as PHP) to allow programmers to access databases contained within a server (or other servers via a network) to be used within web-based applications via the web-browser.

We won't spend much time learning the details of MySQL configuration but you need a basic server set up. You may remember when setting up MySQL from OPS245 - it is basically the same concept.

MySQL / SQL Language Resources

Again, MySQL can be a complex topic: Seneca has an entire course that concentrates on using SQL commands! Here is a link to MySQL / SQL Language resources: MySQL / SQL Language Resources.

Perform the following steps:

1. Install **mariadb-server**.
    a. The MySQL and MariaDB (made by the original developers of MySQL) are actually two separate projects run by the same group, yet they are compatible; therefore, you can use documentation from one to configure the other.

2. When you start the MySQL service, check the system log file for instructions regarding how to set the root password. Even though we will not configure our MySQL service to be **accessible over the network**, it is accepted as a "best practice" configuring for network access for each MySQL installation.
3. Make sure you **start** and **enable** the **mariadb** service. Then refer to the log file (by running **journalctl -xe**) to learn how to run the two commands in order to generate the appropriate passwords.
   a.

```
-- Support: http://lists.freedesktop.org/mailman/listinfo/systemd-devel
--
-- Unit mariadb.service has begun starting up.
Nov 16 16:58:40 vm1 mariadb-prepare-db-dir[11403]: Initializing MariaDB database
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: 221116 16:58:41 [Note] /usr/libexec/mysqld (mysqld 5.5.68-MariaDB) starting as process 11470 ...
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: 221116 16:58:41 [Note] /usr/libexec/mysqld (mysqld 5.5.68-MariaDB) starting as process 11479 ...
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: PLEASE REMEMBER TO SET A PASSWORD FOR THE MariaDB root USER !
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: To do so, start the server, then issue the following commands:
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: '/usr/bin/mysqladmin' -u root password 'new-password'
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: '/usr/bin/mysqladmin' -u root -h vm1 password 'new-password'
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: Alternatively you can run:
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: '/usr/bin/mysql_secure_installation'
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: which will also give you the option of removing the test
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: databases and anonymous user created by default.  This is
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: strongly recommended for production servers.
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: See the MariaDB Knowledgebase at http://mariadb.com/kb or the
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: MySQL manual for more instructions.
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: Please report any problems at http://mariadb.org/jira
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: The latest information about MariaDB is available at http://mariadb.org/.
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: You can find additional information about the MySQL part at:
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: http://dev.mysql.com
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: Consider joining MariaDB's strong and vibrant community:
Nov 16 16:58:41 vm1 mariadb-prepare-db-dir[11403]: https://mariadb.org/get-involved/
Nov 16 16:58:41 vm1 mysqld_safe[11485]: 221116 16:58:41 mysqld_safe Logging to '/var/log/mariadb/mariadb.log'.
Nov 16 16:58:41 vm1 mysqld_safe[11485]: 221116 16:58:41 mysqld_safe Starting mysqld daemon with databases from /var/lib/mysql
Nov 16 16:58:43 vm1 systemd[1]: Started MariaDB database server.
-- Subject: Unit mariadb.service has finished start-up
-- Defined-By: systemd
-- Support: http://lists.freedesktop.org/mailman/listinfo/systemd-devel
--
-- Unit mariadb.service has finished starting up.
--
-- The start-up result is done.
Nov 16 16:58:43 vm1 polkitd[642]: Unregistered Authentication Agent for unix-process:11397:79922188 (system bus name :1.727, object path /org/freede
lines 2172-2205/2205 (END)
```

4. mariadb don't have a default password, create the password with either of the following command:
   a. '/usr/bin/mysqladmin' -u root password 'Password1234'
   b. '/usr/bin/mysqladmin' -u root -h hostname password 'Password1234'

NOTE: If you are building a **LAMP** solution for production, use a password you make up yourself, but do not use your own secret password, since you will be storing that password in a plain text file for later reference. We are building our PoC (Proof of Concept) environment, we could use the default 'Password1234'.

5. if your mariadb installation was messed up and getting errors, you can use the following command to do a clean uninstall of mariadb and then repeat from the step 1.

```
yum remove mariadb mariadb-server
rm -rf /var/lib/mysql
rm -rf /etc/my.cnf
```

Test Connection to MySQL Database Server

While the web server (with php), and MySQL server may be working individually, we need to ensure that they can connect to each other. Since this test will involve storing the database password in a plain-text html file, we want to make sure no one else can access it.

Perform the following steps:

1. Modify the Directory statement for your **private** directory to prevent any machine other than your Server VM from accessing it.

```
<Directory "/var/www/html/private">
AllowOverride None
Require local
</Directory>
```

2. Restart the webserver and try to access the page from another machine. Make sure that you **cannot** do so before you continue.
3. Install the **php-mysql** module so that the installation of php your web server is using can execute sql statements. You will have to restart the service after installing it.
4. Modify the index.php page in your private directory to match the code below. This will test that your web server can connect to the database (replace the <user> and <password> with values appropriate for your machine): if your password has **special characters**, you must enclose it with single quotes. *** Do not use single quote inside your password. ***

```
<?php
$mysqli = new mysqli("localhost", "root", "Password1234");
if ($mysqli->connect_errno) {
echo "Failed to connect to MySQL: (" . $mysqli->connect_errno . ") " . $mysqli->connect_error;
}
echo $mysqli->host_info . "\n";
?>
```

5. Check what you get when you run the following from the server vm: lynx http://localhost/private
6. Now try again from Windows VM, are you able to connect? Why not?
7. Copy the same index.php to /etc/www/html/index.php and then try step 5 and step 6.
    a. Remember to remove the index.html from /etc/www/html/, otherwise, index.html will be loaded.
    b. What else you can do without remove the index.html? (change of conf file require restart of service)

```
<IfModule dir_module>
    DirectoryIndex index.php index.html
</IfModule>
```

You have now established that the web server is able to run code which can interact with the database. This will allow dynamic pages to make use of information stored there when providing resources to your users.

Case 3: Install, Configure & Run Webmail Application (Roundcube Mail)

In this case study, we will simply install, configure and run the **roundcube** webmail application. **Perform the following steps on Server VM:**

1. Download the **Complete** "zipped tarball" from their website (https://roundcube.net/download/) or use **wget** on the download link. You may need to install **wget.**

   *** we will use roundcube 1.4.10 as the latest roundcube 1.5.2 need PHP version higher than 5.4 ***

   ```
   wget https://github.com/roundcube/roundcubemail/releases/download/1.4.10/roundcubemail-1.4.10-complete.tar.gz
   ```

2. Extract the "zipped tarball" and rename the generated directory that contains download source code to: webmail. Also make sure that webmail is a sub-directory of your **DocumentRoot**.
   Use the **--no-same-owner** option when extracting the tar archive to ensure that the files do not keep the original owner (who will not exist on your system).

   ```
   tar -xf roundcubemail-1.4.10-complete.tar.gz -C /var/www/html/ --no-same-owner
   mv /var/www/html/roundcubemail-1.4.10/ /var/www/html/webmail/
   ```

3. Change the ownership of the temp and logs directories so they belong to apache.

   ```
   chown apache /var/www/html/webmail/temp/
   chown apache /var/www/html/webmail/logs/
   ```

4. If you have SELinux enabled: This service needs to be able to write to several directories (temp and logs) that SELinux prevents write access to. If you are in a section that has SELinux set to enforcing, run the following commands to let it know that apache should be allowed to write to files in those directories.

```
semanage fcontext -a -t httpd_log_t '/var/www/html/webmail/temp(/.*)?'
semanage fcontext -a -t httpd_log_t '/var/www/html/webmail/logs(/.*)?'
restorecon -v -R /var/www/html/webmail
```

NOTE: If your machine does not have the **semanage** command, use yum to install the **policycoreutils-python** package.

You will also need to tell selinux to allow the webserver to open connections to the MTAs with

```
setsebool -P httpd_can_network_connect 1
```

5. In the directory now named "webmail", there will be a file named INSTALL which will walk you through the rest of the Roundcube installation.

Some installation tips to consider:

   a. Be careful about copying & pasting the MySQL setup part: take time and pay attention to detail: do not try to "rush it". Login to MySQL and run the following command.
      i. mysql -u root -p

```
CREATE DATABASE roundcubemail CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
CREATE USER roundcube@localhost IDENTIFIED BY 'Password1234';
GRANT ALL PRIVILEGES ON roundcubemail.* TO roundcube@localhost;
```

   b. Exit MySQL command and you will then need to install additional Apache php supporting modules including: **php-xml** and **php-mbstring**. And remember, restart of **httpd** required.

c. Don't forget to change the password in the roundcube configuration.

cp /var/www/html/webmail/config/config.inc.php.sample /var/www/html/webmail/config/config.inc.php

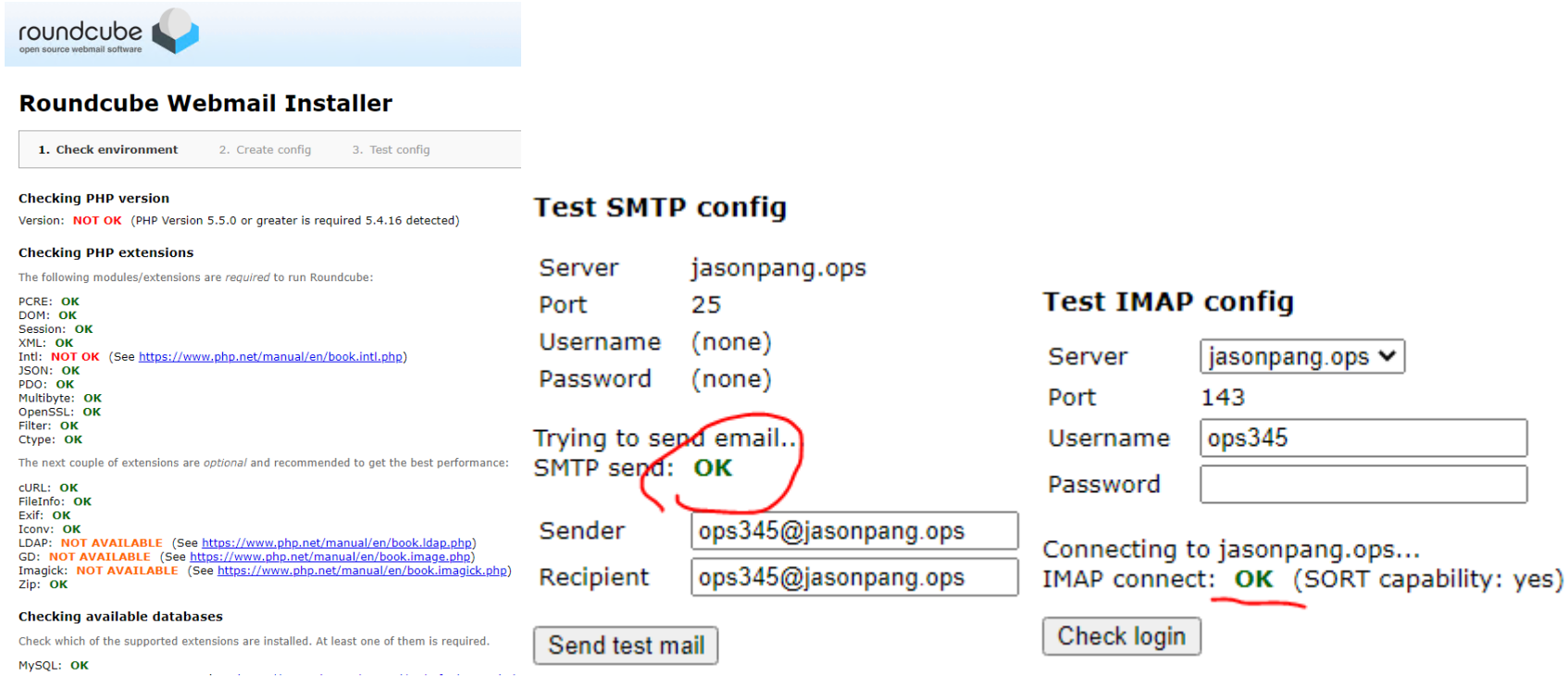
modify the config.inc.php, insert a line or modify existing line with the following: or use **installer** to configure.

```
$config['db_dsnw'] = 'mysql://roundcube:Password1234@localhost/roundcubemail';
$config['enable_installer'] = true;
$config['default_host'] = 'jasonpang.ops';
$config['smtp_server'] = 'jasonpang.ops';
$config['smtp_port'] = 25;
$config['smtp_user'] = '';
$config['smtp_pass'] = '';
```

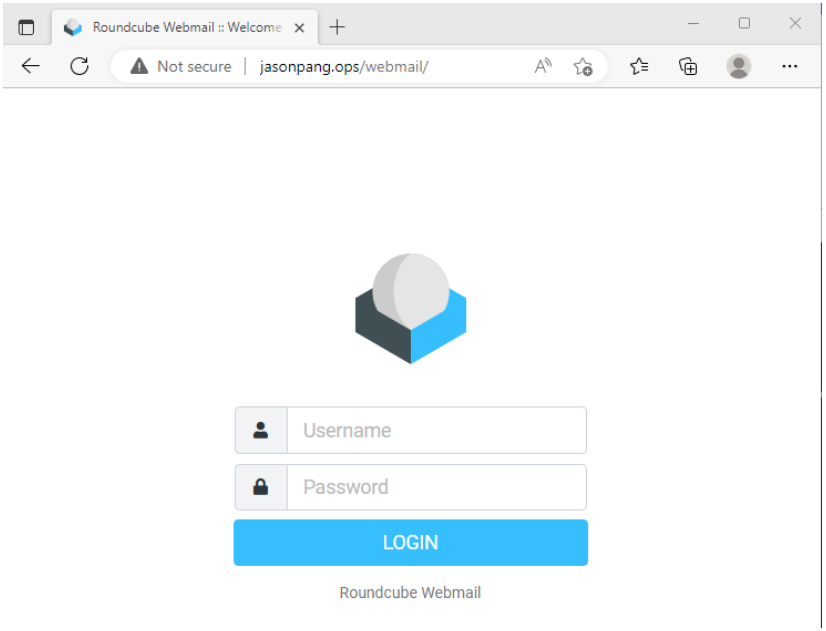
6. To make things easier, RoundCube has a well configured installation page available through your local web browser (You will see a note about it in the INSTALL file).
   a. Go onto your **Windows VM**, open a browser and on the address bar type **server.<yourSenecaID>.ops/webmail/installer**, make sure your DNS on host can resolve the web address. Alternatively, instead of "server.<yourSenecaID>.ops" you can input the ip address of your Server VM, **192.168.0.X/webmail/installer**, change X to your own IP octet.
   b. Inside the web browser installer, ensure all required options are "ok", if "DOM: not ok" it means you need to install additional php packages (yum install php-xml php-mbstring). Once everything is ready (it will not let you continue otherwise) click next go to the next page.
   c. On the next page, everything should be configured if you made the change on /var/www/html/webmail/config/config.inc.php

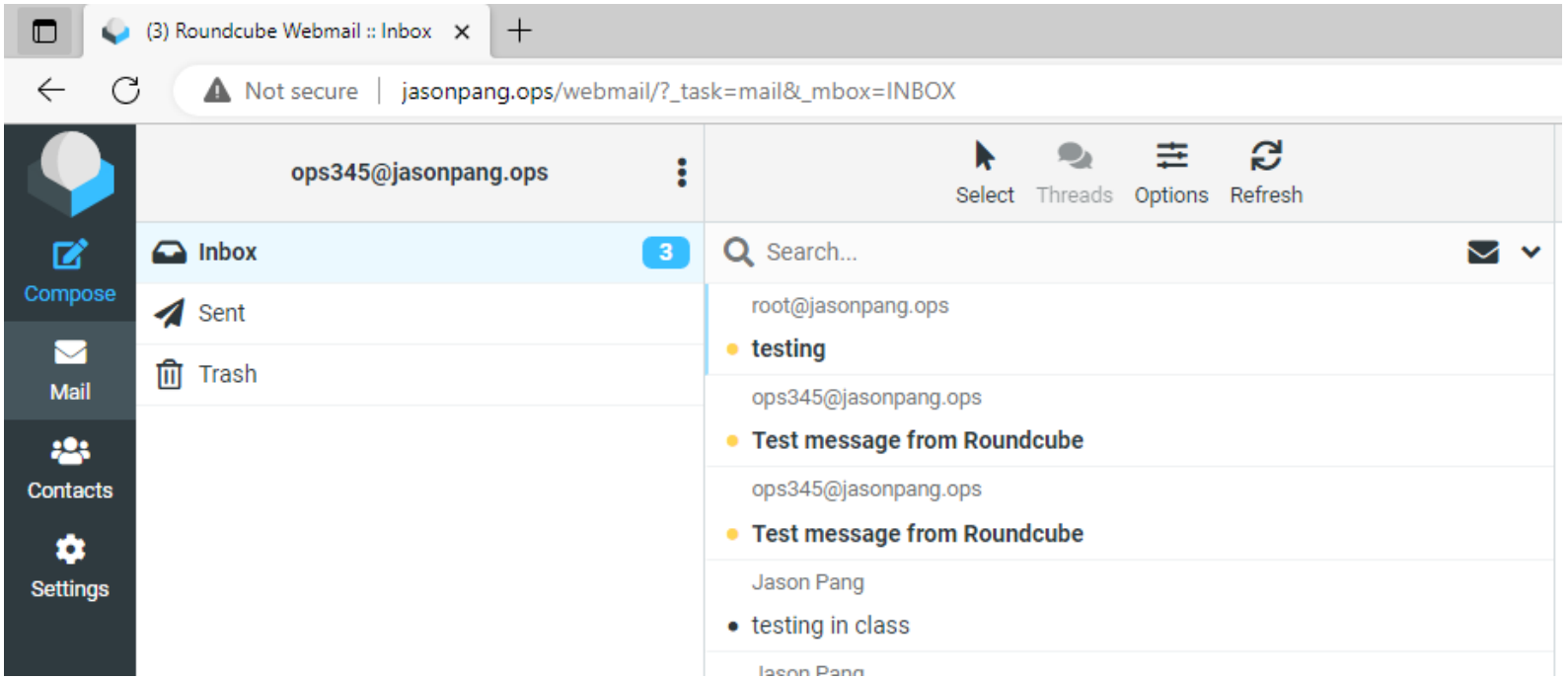
d. If everything configured by config.inc.php, you can skip this section. Under the **IMAP** settings insert **<yourSenecaID>.ops** in **default_host** field and **143** in **default_port** field. Under SMTP settings insert **<yourSenecaID>.ops** in smtp_server field, and **25** in **smtp_port** field. Ensure **smtp_user/smtp_pass** is empty and uncheck the "Use the current IMAP username and password for SMTP authentication" checkbox.
e. Under **Database setup db_dsnw**, enter "**localhost**" as your database server, "**roundcubemail**" for database name. Put "**roundcube**" as Database user, and the password (password1234) you set for the roundcube user when you configured that in the previous step for database password. Everything else can be left as default.
f. Click Next to create the configuration file, then download it to your host. By default, it will be saved under ~yourSenecaID/Downloads. Transfer that file to Server VM using scp and place it inside /var/www/html/webmail/config folder.
g. Go to test config page if you are not there already and "Check config file" should be ok. "Check DB config" should also be ok, if not check your mysql settings.
h. Make sure your SMTP and IMAP servers are running, then finally test your configuration by sending email using your smtp server through test field provided by webmail installer, you should receive a test email sent by RoundCube. Test your IMAP settings by simply loging in with your OPS345 and Password1234 password on the same webpage.

7. Test on the next tab to make sure you get everything ok.



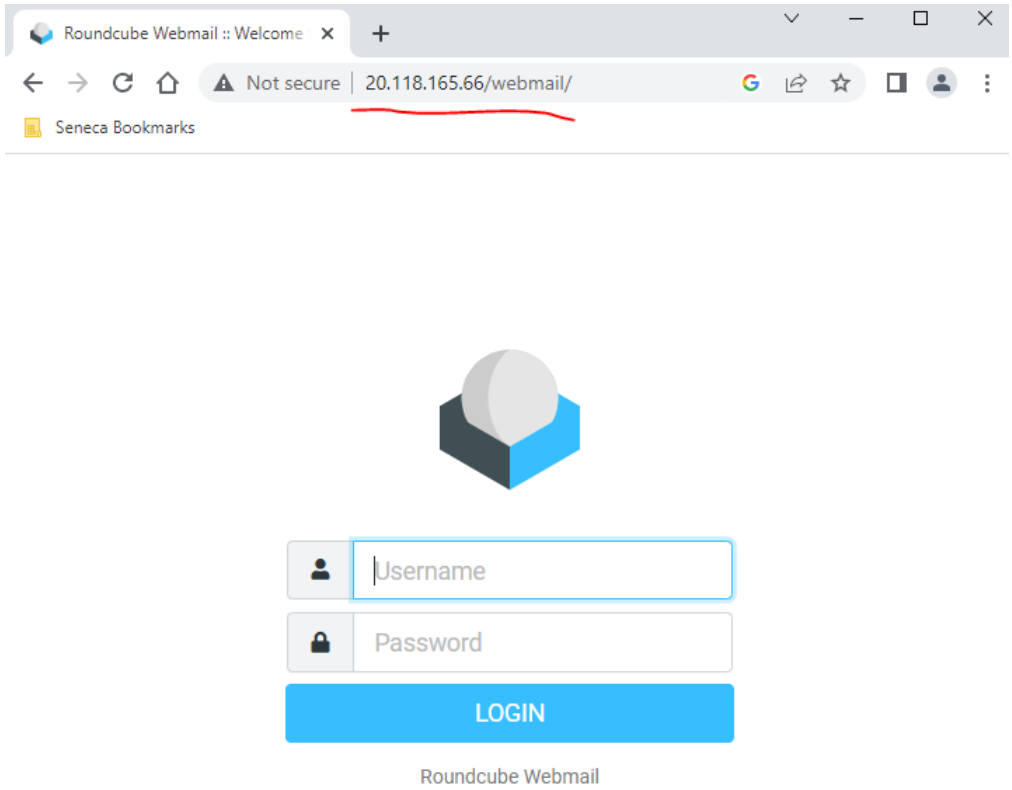
8. Now that you have Roundcube installed it is time to test if the roundcube webmail application is working by logging on, then sending and receiving e-mail messages:
    a. Using the any browser, navigate to server.<yourdomain>.ops/webmail and login (using the username only).
    b. Use the interface provided to send and receive email.


Roundcube Webmail :: Welcome
Not secure | jasonpang.ops/webmail/
Username
Password
LOGIN
Roundcube Webmail


(3) Roundcube Webmail :: Inbox
Not secure | jasonpang.ops/webmail/?_task=mail&_mbox=INBOX
ops345@jasonpang.ops
Select
Threads
Options
Refresh
Compose
Mail
Contacts
Settings
Inbox 3
Sent
Trash
Search...
root@jasonpang.ops
testing
ops345@jasonpang.ops
Test message from Roundcube
ops345@jasonpang.ops
Test message from Roundcube
Jason Pang
testing in class

9. If mail sent through roundcube is sending from the wrong domain (i.e. user@server.yourdomain.ops instead of user@yourdomain.ops), each user can override it in the settings tab, or you can set: $config['mail_domain']

Check if you can access from external devices with http://vm1.public.IP.address/webmail



Once everything tested working. Remove the newly created Windows VM.

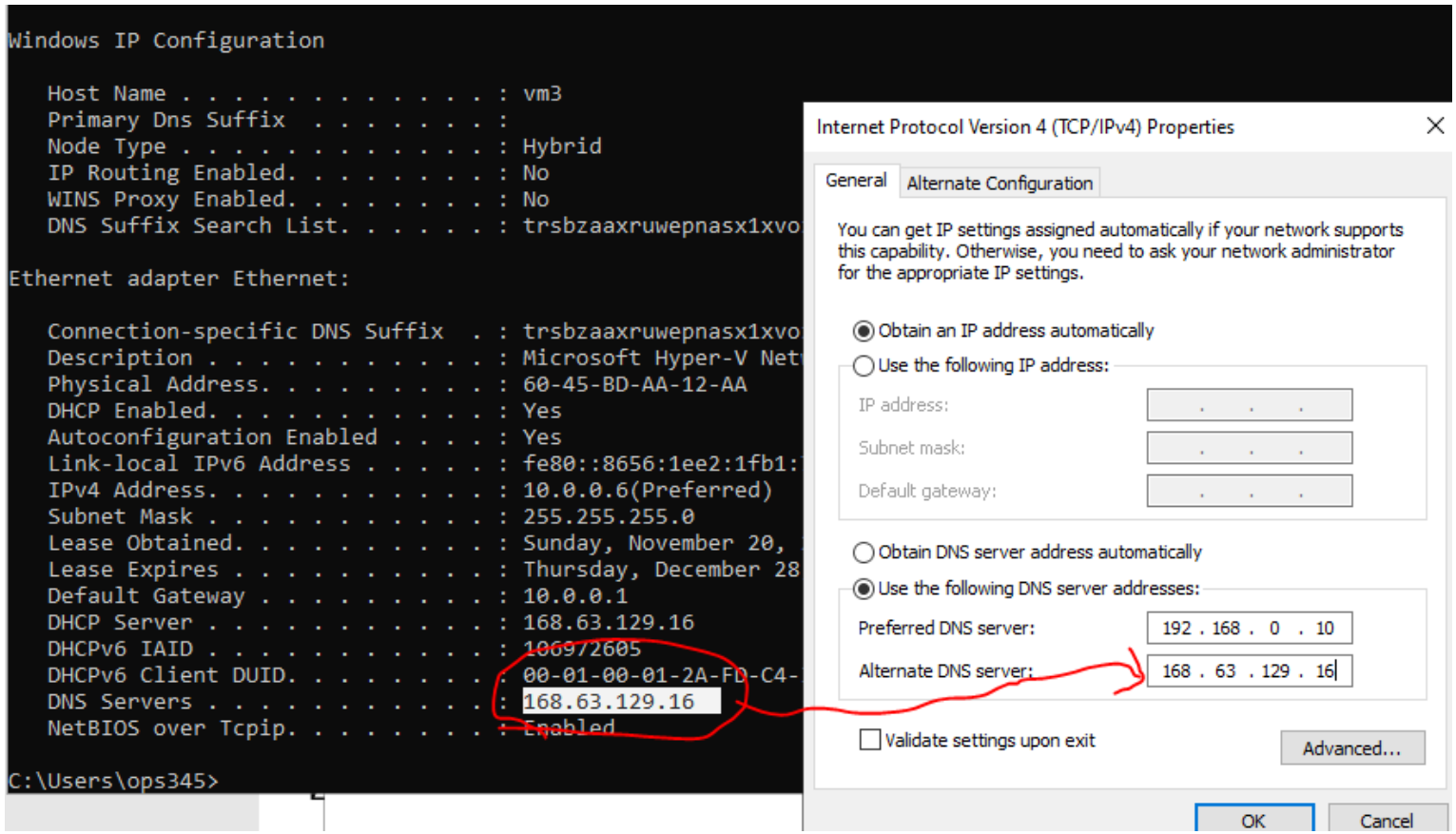
Save the captured file(s) as OPS345_Lab08_yourusername and upload to Blackboard.

If it is video recordings, upload to OneDrive and share with jason.pang@senecacollege.ca

Appendix:

If you have any issues to accomplish the above steps. Check the following.

- Server vm:
  - TCP/UTP port 53, also check if **named** service running.
  - TCP port 80, also check if **httpd** service running.
  - TCP port 25, 143, also check if **postfix** or **dovecot** service running.
- Windows VM:
  - DNS server as 192.168.0.10???



- Other command to try
  - systemctl restart postfix
  - systemctl restart httpd
  - systemctl restart mariadb
- From the first day of our class, we configured all services to run at 192.168 network. Check if your Windows VM has such network.