Introduction:

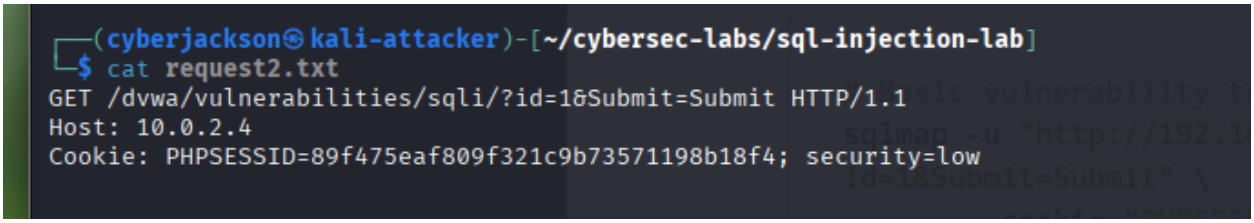
SQL Injection is a cybersecurity activity I have heard of since I began studying. Now I finally get to attempt SQL injection myself in this lab. It is one of the most critical web application security vulnerabilities and for cybersecurity professionals understanding it is essential for pen testing, security assessments, and incident response. SQL Injection allows attackers to manipulate database queries, potentially leading to data theft, authentication bypass, and complete system compromise.

By the end of this lab, I will be able to understand basic SQL database and syntax, identify SQL injection vulnerabilities, execute manual SQL injection attacks, utilize SQLMap for automated exploitation, extract sensitive information from database, and assess business impact of SQL injection vulnerabilities. In the end, I will tie this all together to apply professional penetration methodology to database security.

Also, we collect the password hash for each user

Exercise: Automated SQL Injection with SQLMap

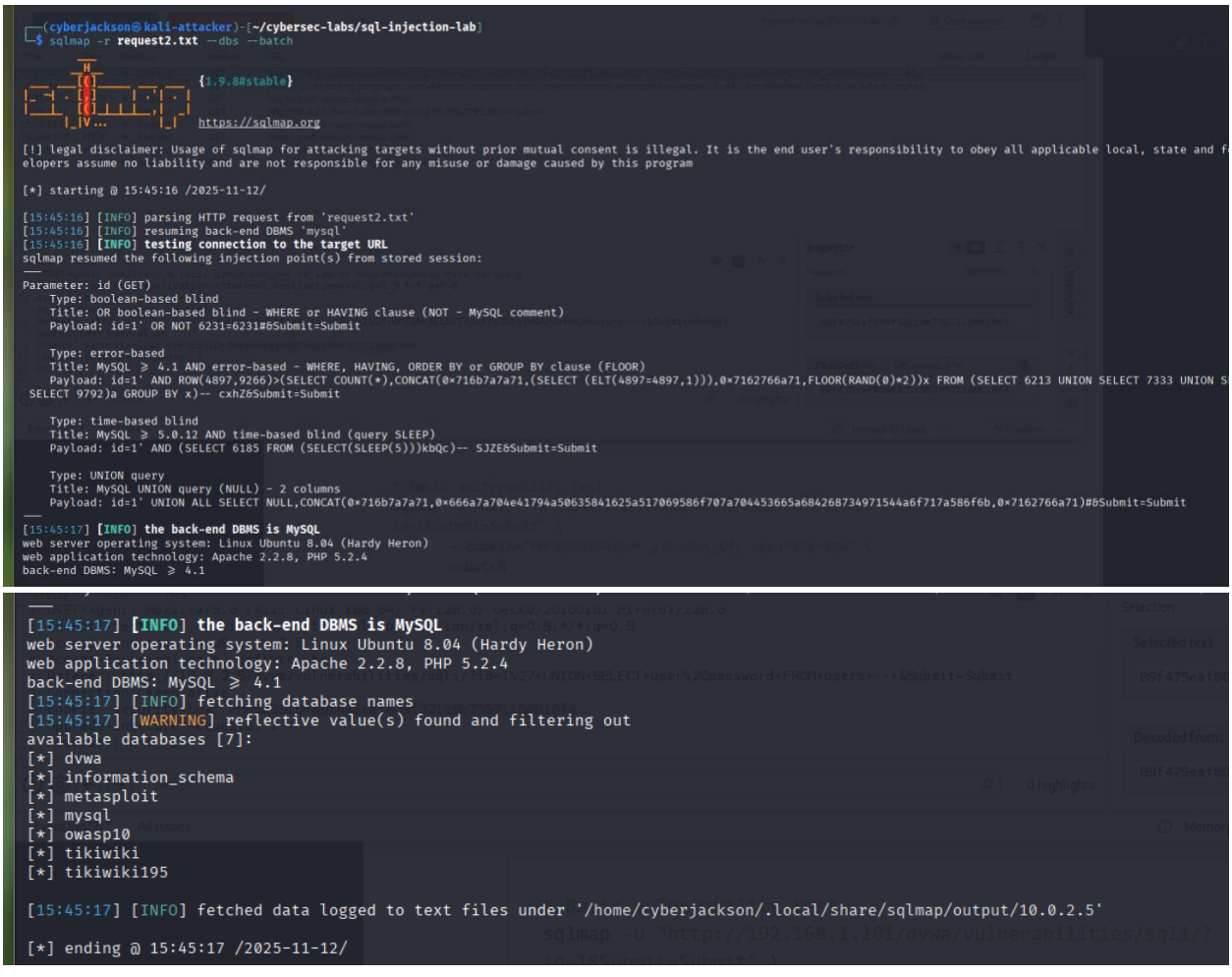
Basic SQLMap usage, creating request.



SQLMap vulnerability detections, running request

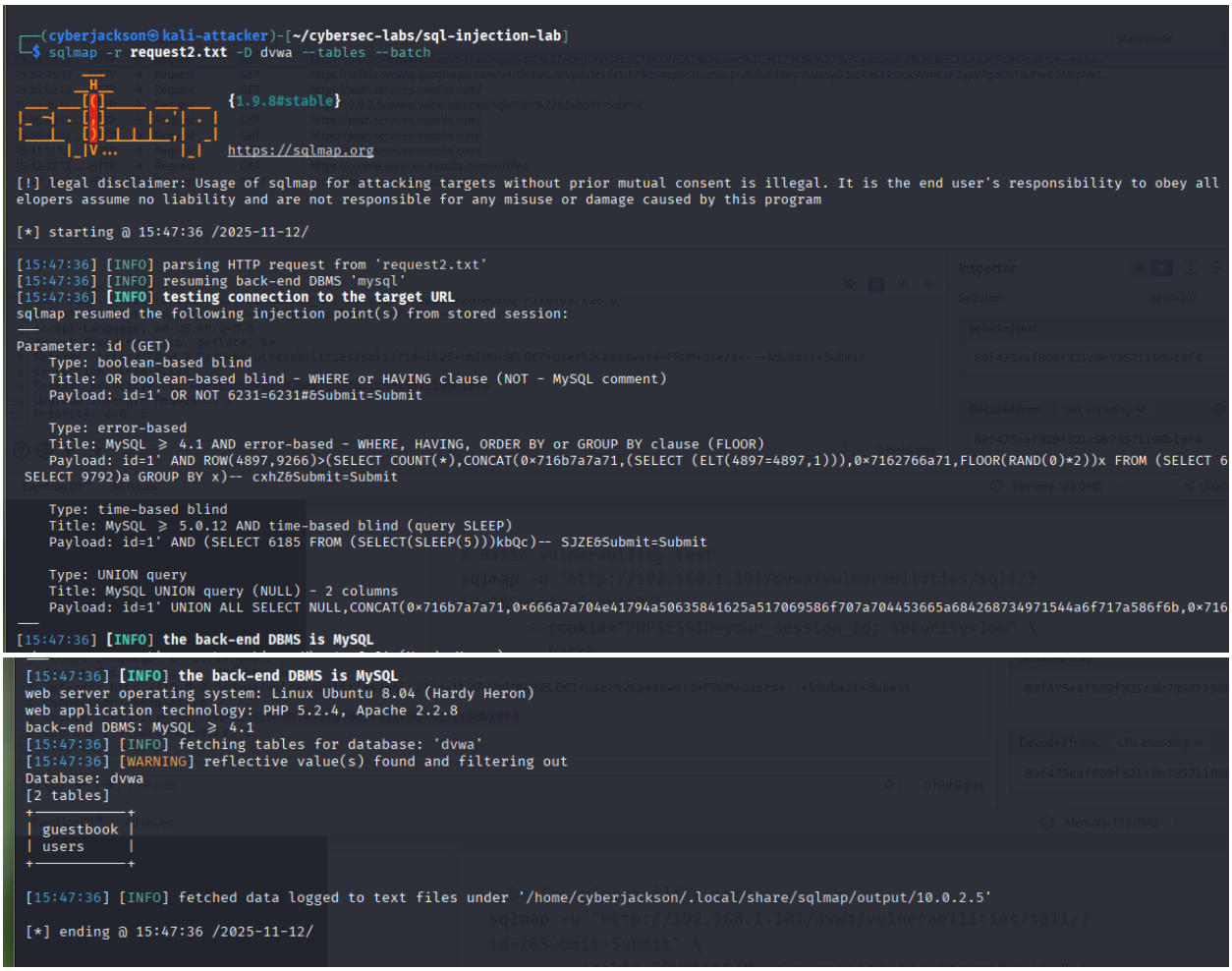
**Note it is doing similar vulnerability scanning as we did earlier in the lab

Database enumeration with SQLMap



Found 7 databases

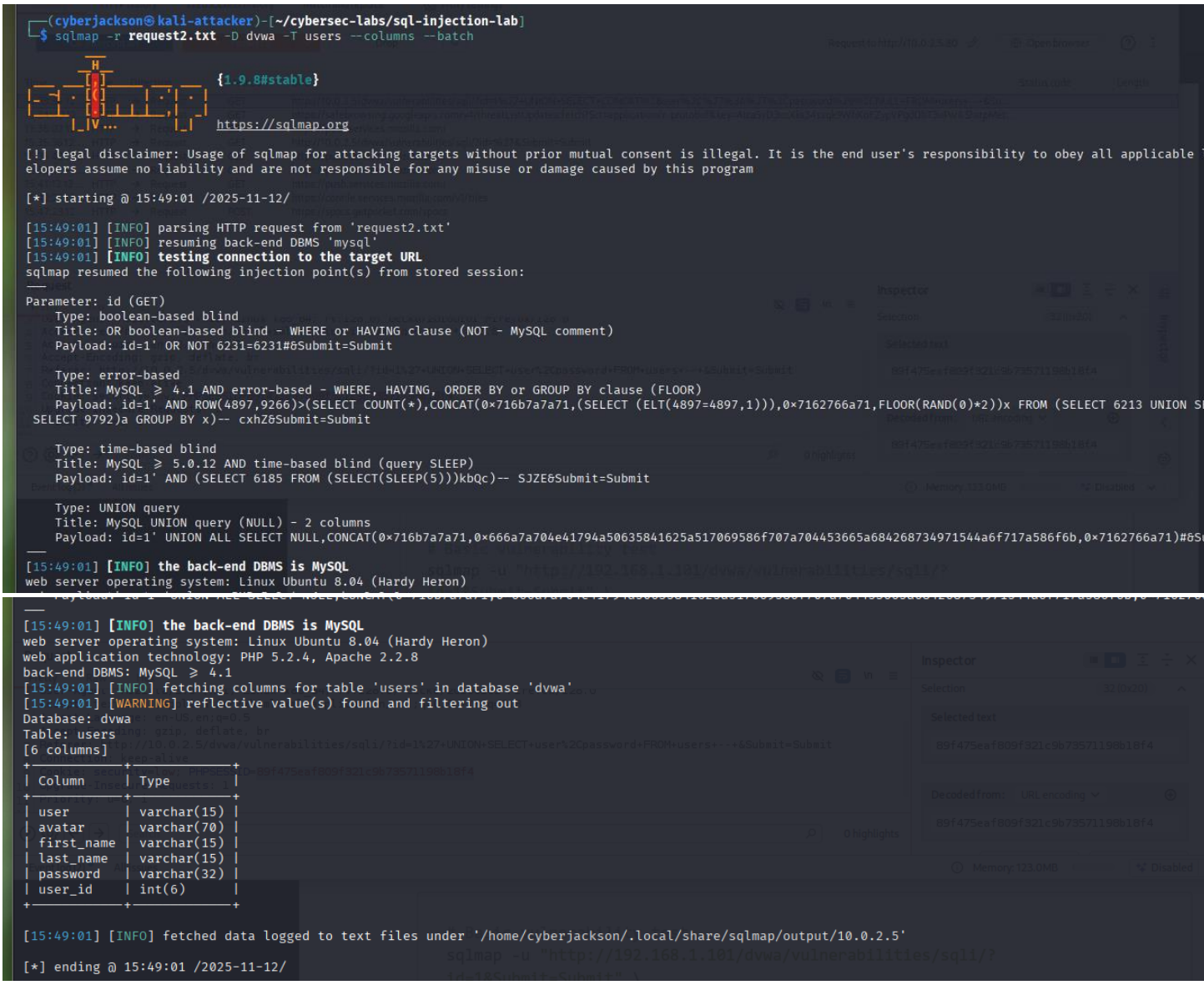
Get tables in current database



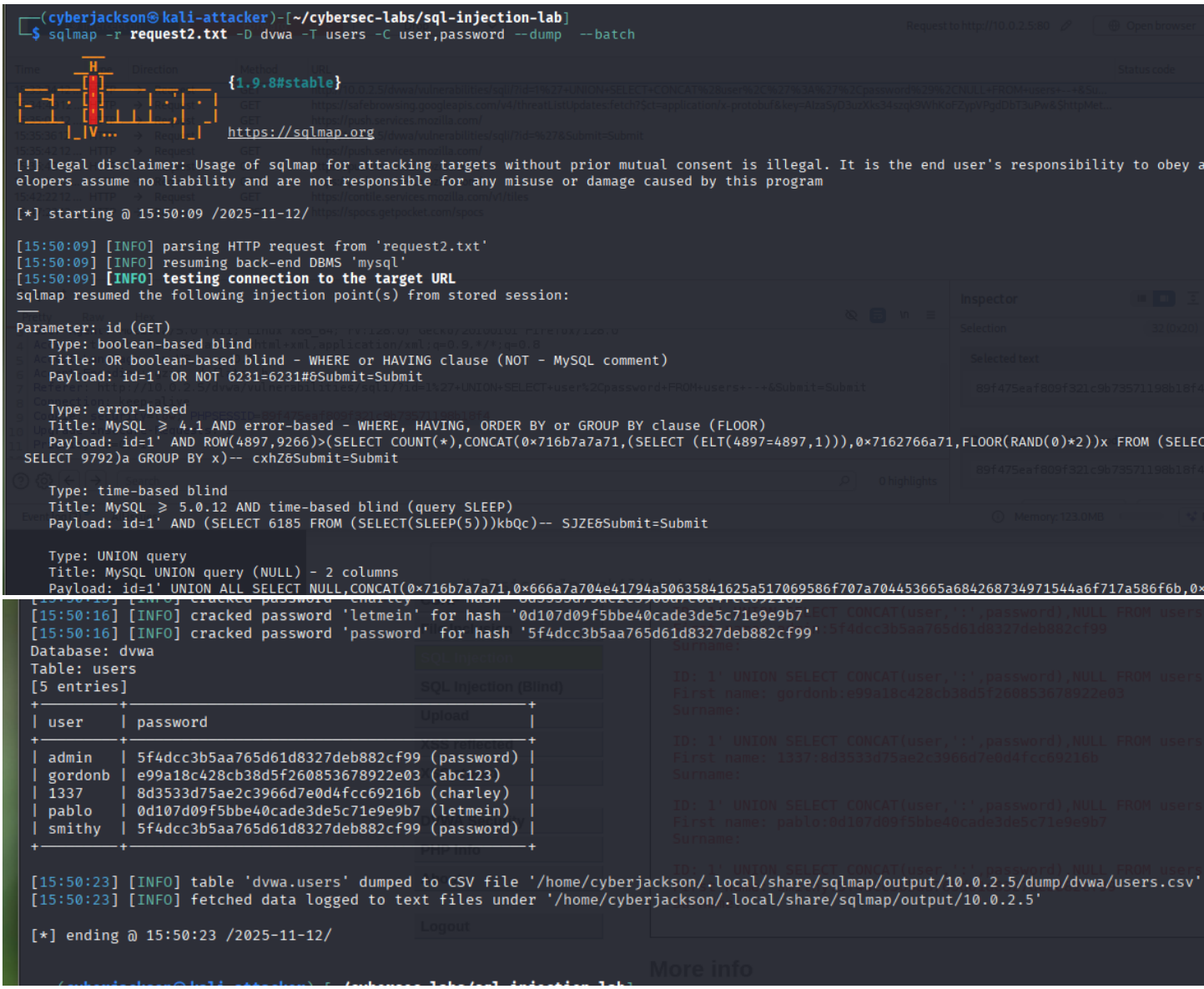
Found the guestbook and users tables in the dvwa database

Data extraction with SQLMap:

Getcolumns in users table



Dump specific columns

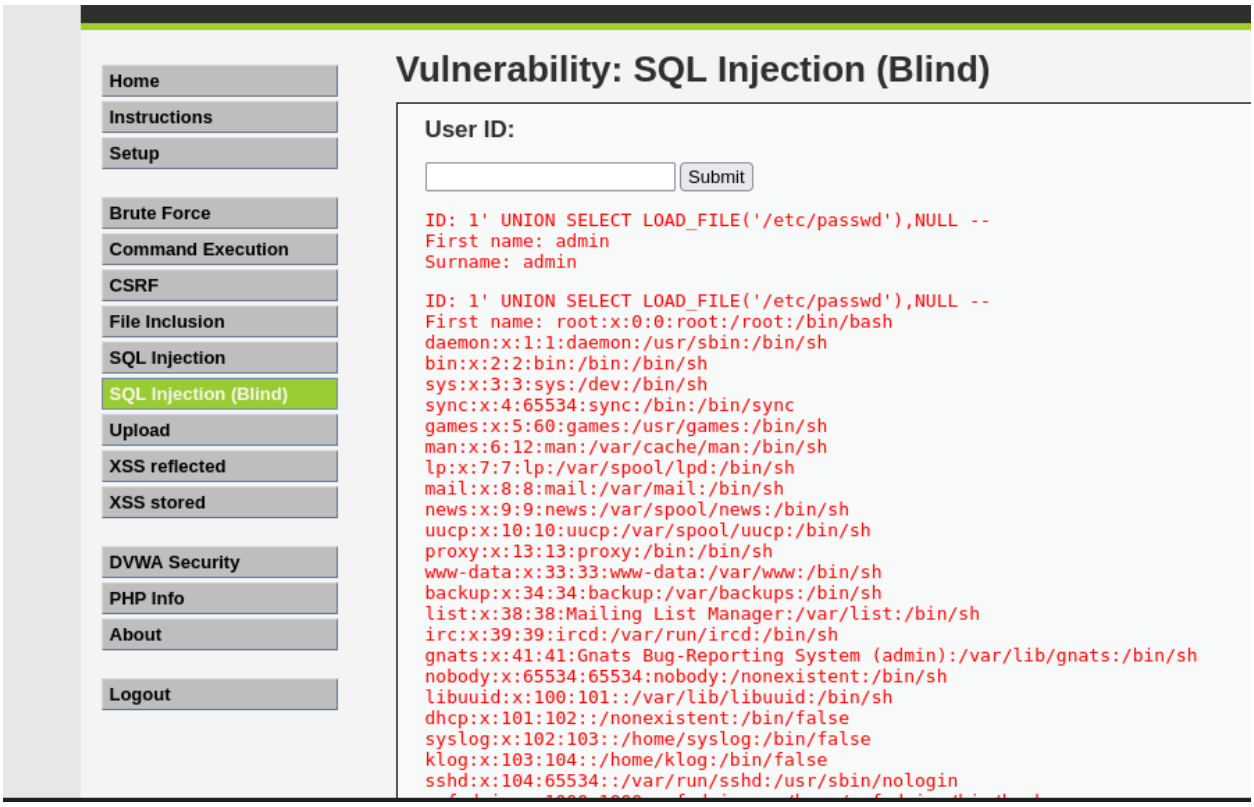


***Note: It actually cracked the password hashes!!!!!

Excercise: Advanced Injection Techniques

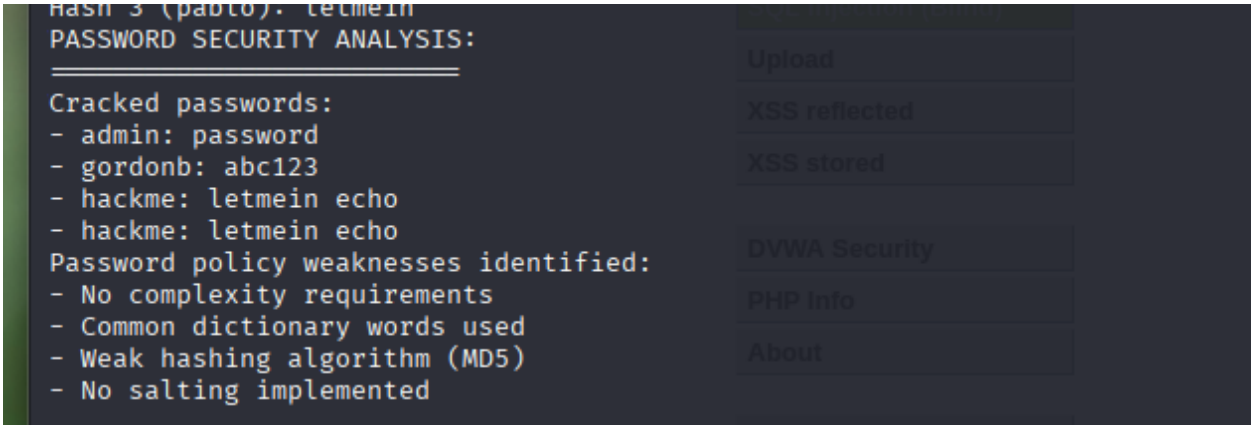
Database specific payloads

Input: 1' UNION SELECT LOAD_FILE('/etc/passwd'),NULL --



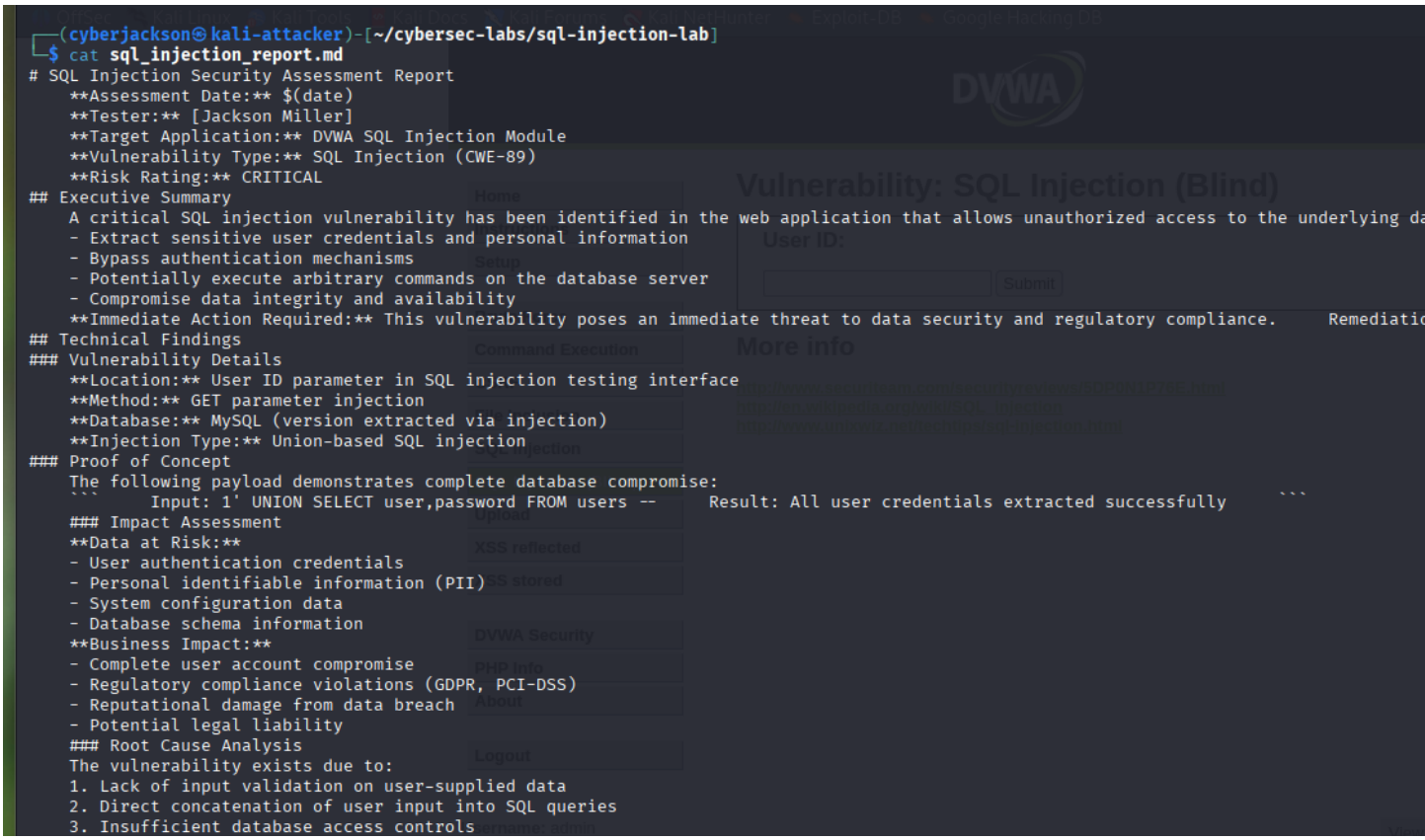
Shows us the file paths of the host machine

With: 1' UNION SELECT 'shell code',NULL INTO OUTFILE '/var/www/shell.php' - you can create a shell attached to the host machine.



Exercise 8: Professional Reporting and Remediation

Create executive summary



```
┌──(cyberjackson㉿kali-attacker)-[~/cybersec-labs/sql-injection-lab]
└─$ cat sql_injection_report.md
# SQL Injection Security Assessment Report
    **Assessment Date:** $(date)
    **Tester:** [Jackson Miller]
    **Target Application:** DVWA SQL Injection Module
    **Vulnerability Type:** SQL Injection (CWE-89)
    **Risk Rating:** CRITICAL
## Executive Summary
    A critical SQL injection vulnerability has been identified in the web application that allows unauthorized access to the underlying da
    - Extract sensitive user credentials and personal information
    - Bypass authentication mechanisms
    - Potentially execute arbitrary commands on the database server
    - Compromise data integrity and availability
    **Immediate Action Required:** This vulnerability poses an immediate threat to data security and regulatory compliance.     Remediatio
## Technical Findings
### Vulnerability Details
    **Location:** User ID parameter in SQL injection testing interface
    **Method:** GET parameter injection
    **Database:** MySQL (version extracted via injection)
    **Injection Type:** Union-based SQL injection
### Proof of Concept
    The following payload demonstrates complete database compromise:
    ```      Input: 1' UNION SELECT user,password FROM users --     Result: All user credentials extracted successfully      ```
    ### Impact Assessment
    **Data at Risk:**
    - User authentication credentials
    - Personal identifiable information (PII)
    - System configuration data
    - Database schema information
    **Business Impact:**
    - Complete user account compromise
    - Regulatory compliance violations (GDPR, PCI-DSS)
    - Reputational damage from data breach
    - Potential legal liability
    ### Root Cause Analysis
    The vulnerability exists due to:
    1. Lack of input validation on user-supplied data
    2. Direct concatenation of user input into SQL queries
    3. Insufficient database access controls
    4. Absence of parameterized queries/prepared statements
    ## Remediation Recommendations
    ### Immediate Actions (24-48 hours)
    1. **Input Validation:** Implement strict input validation for all usersupplied data
    2. **Parameterized Queries:** Replace dynamic SQL with prepared statements
    3. **Database Permissions:** Apply principle of least privilege to database accounts
    4. **WAF Deployment:** Deploy web application firewall as temporary protection
    ### Short-term Actions (1-2 weeks)
    1. **Code Review:** Conduct comprehensive security code review
    2. **Security Testing:** Implement automated security testing in CI/CD pipeline
    3. **Error Handling:** Implement proper error handling to prevent information disclosure
    4. **Monitoring:** Deploy database activity monitoring and alerting
    ### Long-term Actions (1-3 months)
    1. **Security Training:** Provide secure coding training for development team
    2. **SDLC Integration:** Integrate security requirements into development lifecycle
    3. **Regular Testing:** Establish regular penetration testing schedule
    4. **Security Architecture:** Review and enhance overall security architecture
    ## Technical Appendix
    ### Tools Used
    - Burp Suite Professional
    - SQLMap automation framework
    - Manual testing techniques
    - Hash cracking utilities
    ### Evidence Files
    - Extracted database contents
    - HTTP request/response traces
    - SQLMap output logs
    - Password hash analysis results
     EOF
```
```

Conclusion:

This lab was a hands on and in depth delve into SQL injections. I had a lot of fun with the lab and I found the penetration testing very interesting. After completing this lab I can now understand fundamental SQL database concepts, identify SQL injection vulnerabilities, execute both manual and automated attacks, utilize SQLMap for comprehensive vulnerability assessment, extract database information, assess business impact, apply professional pen testing methodology to database security assessment, and finally document findings and implement secure coding practice to harden databases.

My favorite part of this lab was utilizing SQLMap. It was very exciting when the tool went ahead on its own to crack the five users' password hashes. In all I feel this was another great piece added to my introductory penetration testing/ethical hacking knowledge. SQL Injection being one of the OG vulnerabilities it is about time that I give it a go myself.