# SEC320 GROUP 7 PROJECT FINAL REPORT

## ATTACKS AND DETECTION BY

**NAME: KUNJAN SHARMA**

## Attack 1: TCP Port: SYN Scan

**Overview:**

By exploiting the TCP three-way handshake, a TCP SYN scan attack can find open ports on a target machine. SYN-ACK for open ports, RST for closed ports, or no response for filtered ports are the responses that the attacker examines after sending SYN packets to the target ports. The handshake is a covert spying technique since it is kept unfinished to prevent detection.

**Tool Used:**

**Nmap**: A flexible network scanning tool that is frequently used for security audits and network discovery. Nmap can determine the target machine's operating system, open ports, and services that are using those ports. It is preferred due to its extensive feature set and capacity for thorough scans.


**Why did we used Nmap:**

We used Nmap because Nmap offers useful information about computers and networks, we use it for network reconnaissance and security evaluations. It can uncover vulnerabilities or misconfigurations by identifying operating systems, open ports, and services that are currently executing. Nmap assists in determining a system's attack surface by collecting information such as OS details, service versions, and possible vulnerabilities. It is a crucial tool for system and network auditing since it is frequently used in penetration testing, network troubleshooting, and assessing the efficacy of security measures like firewalls or intrusion detection systems.

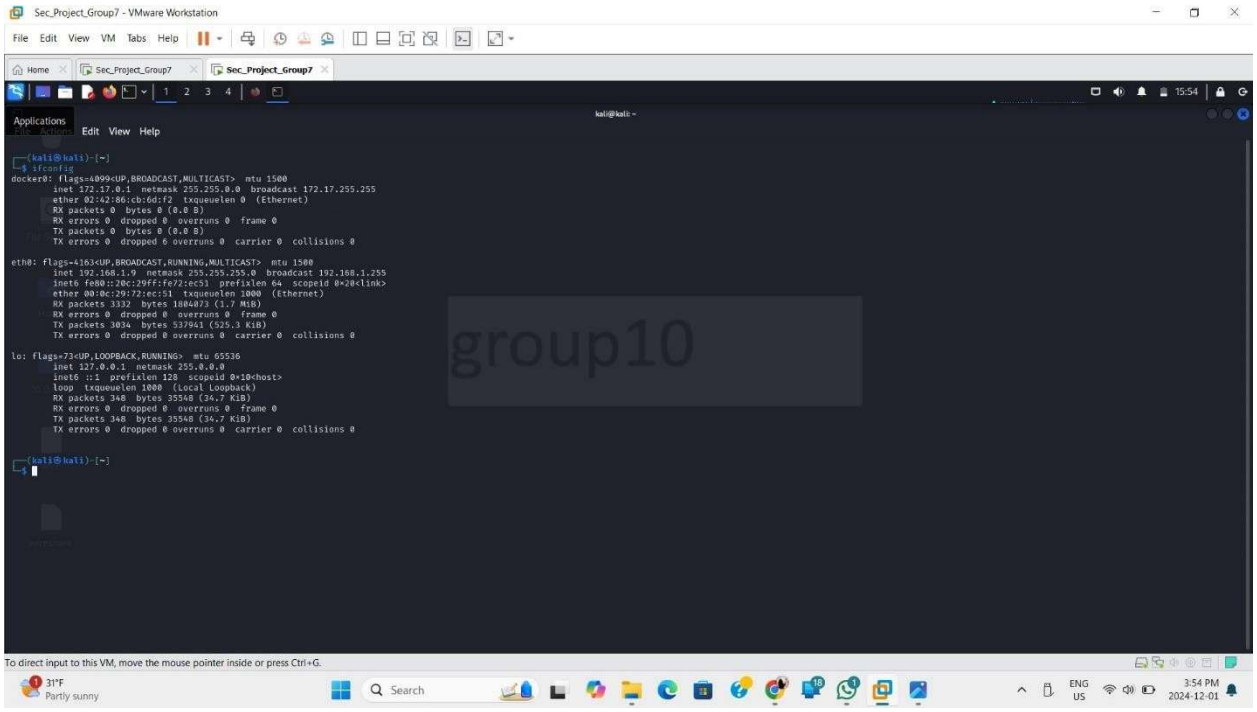
**Network Configuration:**

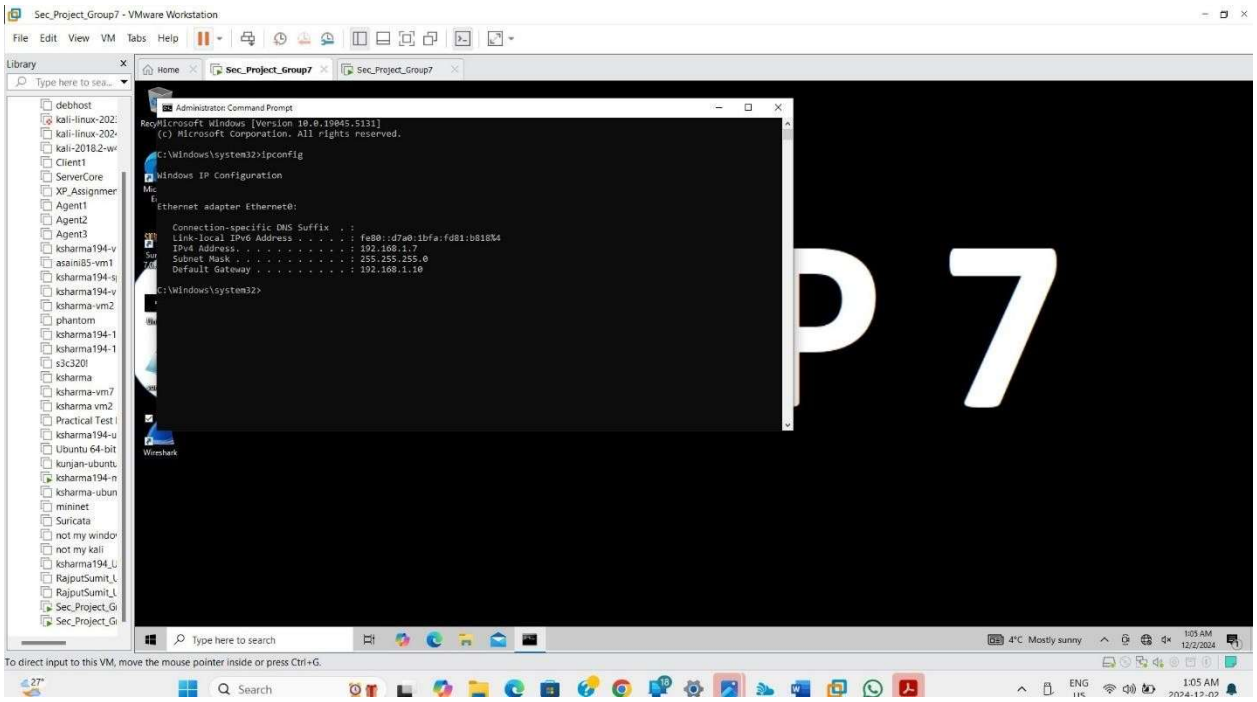
The attack was conducted in a virtualized environment using VMware Workstation, consisting of the following three machines:

- **Attacker**: Kali Linux (192.168.1.9)

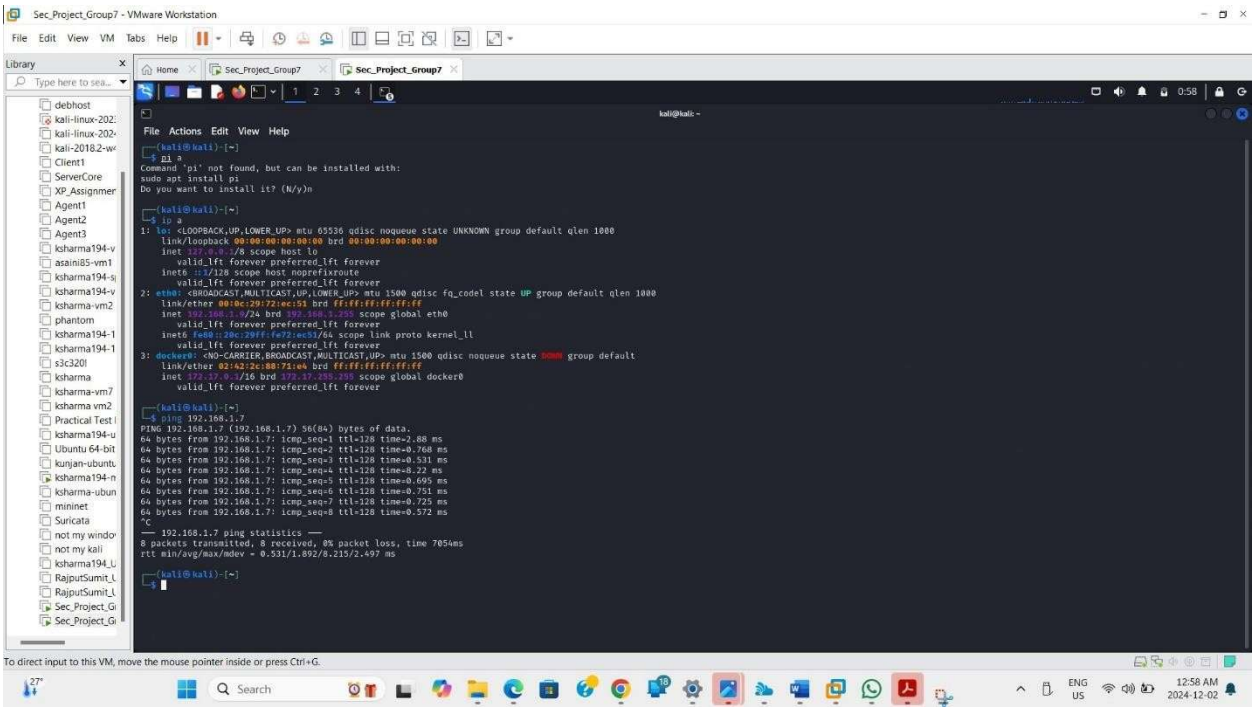
# SEC320 GROUP 7 PROJECT FINAL REPORT

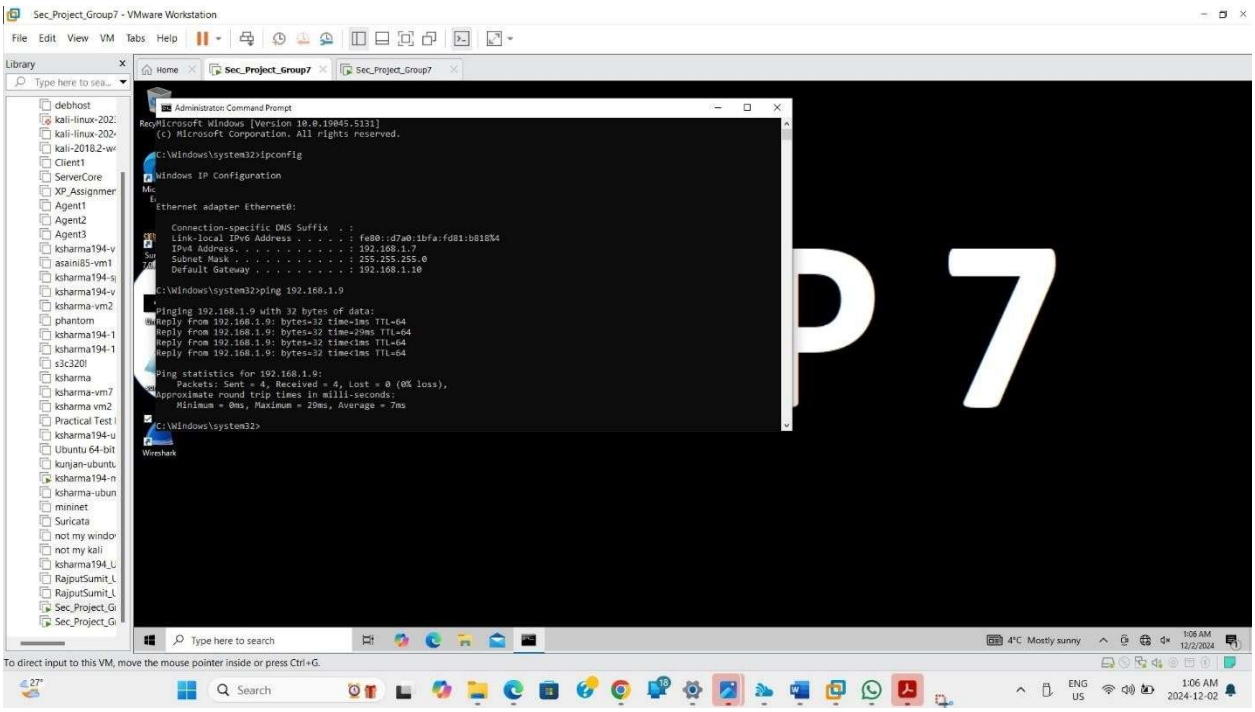
- **Victim**: Windows 10 (192.168.1.7)
- **Observer**: Suricata VM (192.168.1.10)



# SEC320 GROUP 7 PROJECT FINAL REPORT



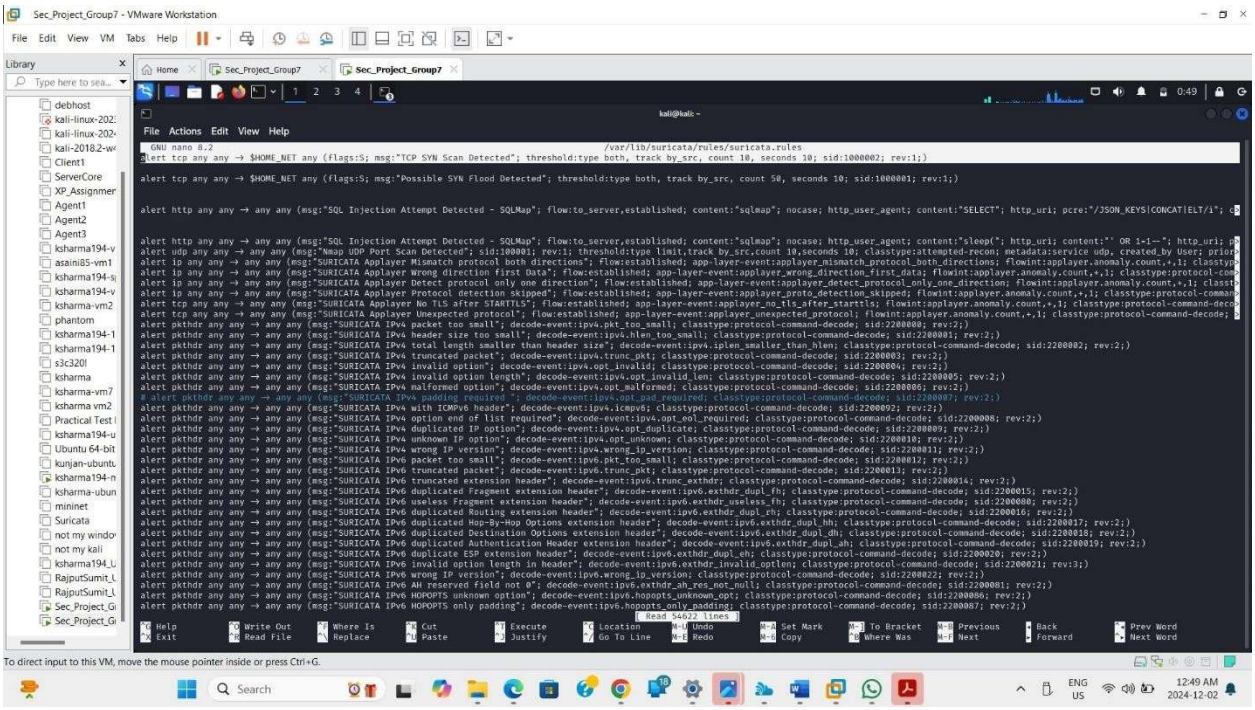




## Preparation:

### File Configuration:

We configered the file sudo nano /var/lib/suricatta/rules/suricatta. Rules to add the the TCP:SYN SCAN attack. This allows suricatta to detect and log the attack.

# SEC320 GROUP 7 PROJECT FINAL REPORT




## Attack Execution:

- The attack was executed using Nmap.
- Command used: `nmap -sS -A -T5 --min-rate 1000 --max-retries 0 -p- --reason --open -v -Pn 192.168.1.7`

**Core Flags:**

- -sS: Performs a SYN scan (half-open scan).
- -A: Enables OS detection, version detection, script scanning, and traceroute.

**Aggressiveness and Speed:**

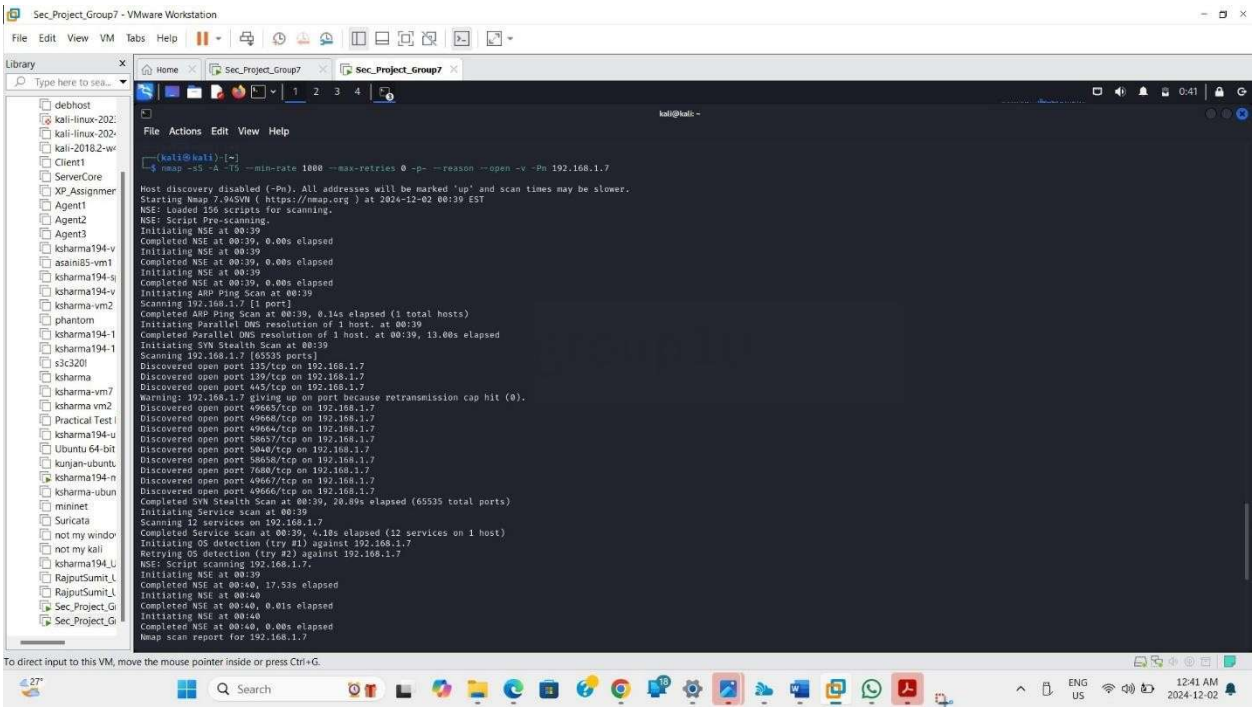
- -T5: Maximum speed timing template. It minimizes delays and aggressively sends packets as quickly as possible.
- --min-rate 1000: Ensures a minimum rate of 1000 packets per second, speeding up the scan.
- --max-retries 0: Completely disables retries for packets, reducing scan time but risking missed responses.

**Comprehensive Scanning:**

- -p-: Scans all 65,535 TCP ports.
- -Pn: Skips the ping/host discovery phase and assumes the host is up.

**Noisy and Obvious:**

- --reason: Displays reasons for each port's status in the output, showing detailed response data from the target.
- --open: Only shows open ports in the output, making it easier to focus on results.
- -v: Increases verbosity for detailed output during the scan.




## TCP SYN Scan's Effect on the Victim:

### Enhanced Network Traffic:

Incoming TCP SYN packets on the victim's system increase, which could overload the network if the scan is aggressive or targets a large number of ports.

### Consumption of Resources:

The system uses resources to manage the TCP handshake for each SYN packet it receives (for example, by establishing half-open connections in the connection queue), which might put a burden on the system when it is being heavily scanned.

### Open Port Exposure:

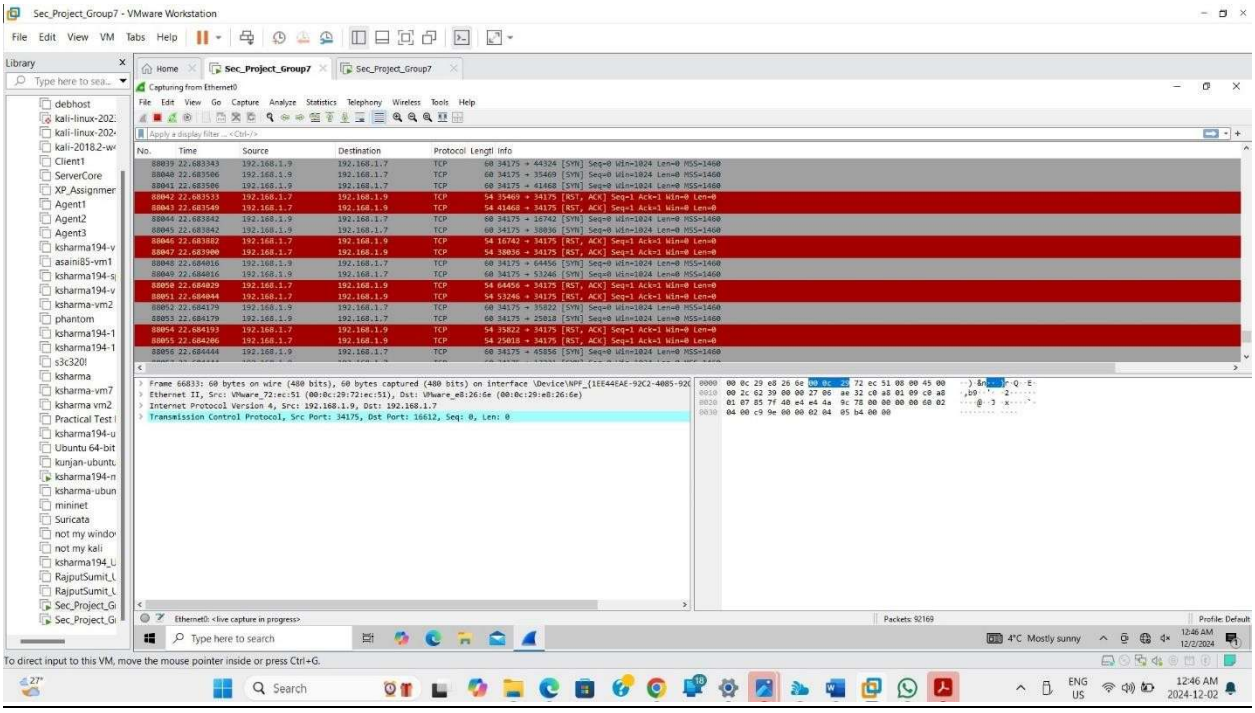
Potential vulnerabilities are revealed to the attacker by the scan, which shows the victim's open ports and active services.

## Observations:

# SEC320 GROUP 7 PROJECT FINAL REPORT

A spike in TCP SYN packets aimed at different ports occurs on the victim computer during a TCP SYN scan. Since no ACK packets follow the SYN-ACK responses, these packets, which are visible in Wireshark, are a component of an unfinished handshake. Here are some important findings: ○ SYN-ACK responses are used to identify open ports, whereas RST responses are used to identify closed ports.

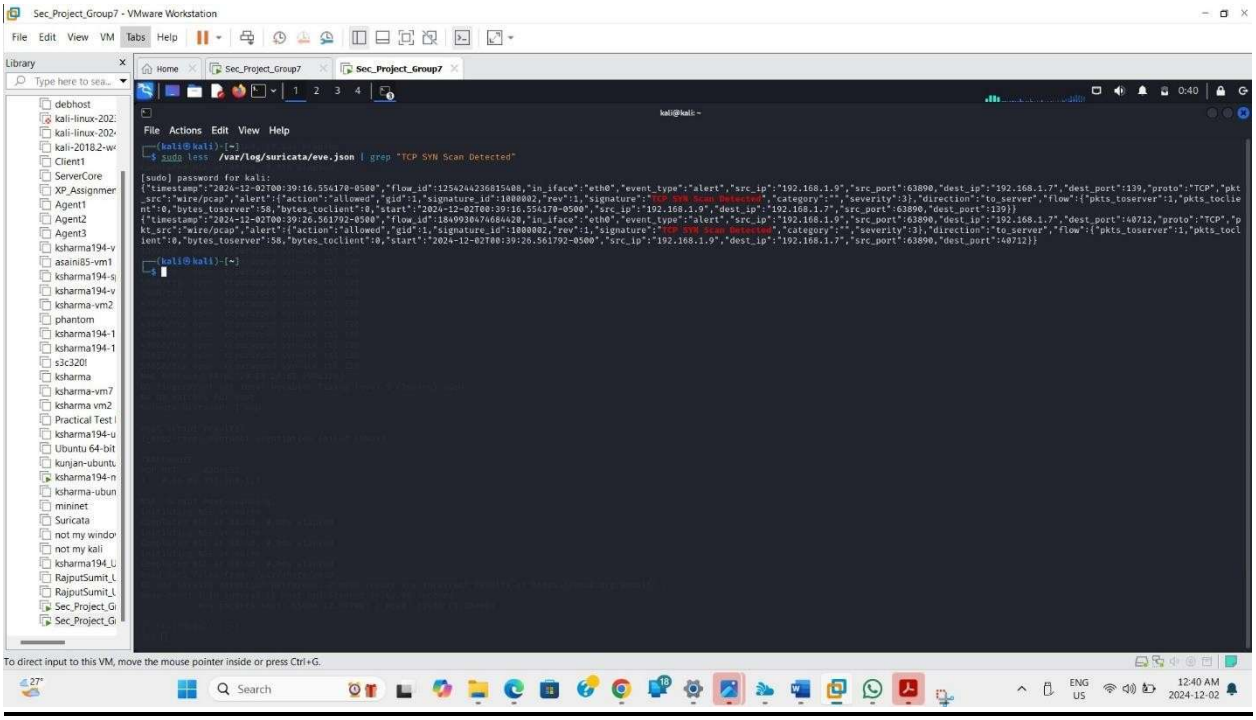
- Depending on the firewall settings, blocked ports produce either ICMP error messages or no response.
- As an indicator of port scanning, Wireshark records high-frequency SYN packets with consistent source IPs and different destination ports.
- The timing and packet rate can be used to determine how aggressive the scan is, and patterns such as random or sequential port probes are frequently seen.




## Detection and Evidence:

We have detected the attack using suricatta

# SEC320 GROUP 7 PROJECT FINAL REPORT




## Protective Measures:

The following preventative actions can be taken to guard against TCP SYN scans, which are used by attackers to find open ports and services:

Set up a firewall:

Employ firewalls to limit access to only reliable IP addresses and prohibit unused ports.
To lower the number of SYN packets permitted from a single IP, enable rate-limiting. IDS/IPS (intrusion detection/prevention systems) should be enabled.

Put SYN cookies into practice:
To avoid the TCP connection queue being exhausted during half-open connections brought on by SYN scans, utilize SYN cookies.

Turn Off Unused Services:
Reduce the attack surface by shutting off unused ports and services.

Employ port knocking.
Use port-knocking strategies, which render scanning useless by keeping ports closed until they are accessed in a particular order.

# SEC320 GROUP 7 PROJECT FINAL REPORT

Turn on monitoring and logging:
Keep an eye on logs for odd activity, including large numbers of SYN packets aimed at several ports, and look into any questionable trends.

Restrict ICMP Reactions:
Set up the system to reduce or prevent ICMP error packets, which can provide hackers more information for espionage.

**Summary Table for the DHCP Spoofing Attack:**

| Attack Name | Launched (formula) | Indicator1 | Indicator2 | Indicator3 | Indicator4 | Possible Tools |
|---|---|---|---|---|---|---|
| TCP SYN SCAN | Launched using Nmap with nmap -sS -A -T5 --min-rate 1000 --max-retries 0 -p- --reason -open -v -Pn 192.168.1.7 | High volume of SYN packets | Rapid port scanning activity | Increased resource usage due to halfopen connections | Logs showing scanning attempts and high packet rate | Nmap |

**Indicator of conversion table:**

# SEC320 GROUP 7 PROJECT FINAL REPORT

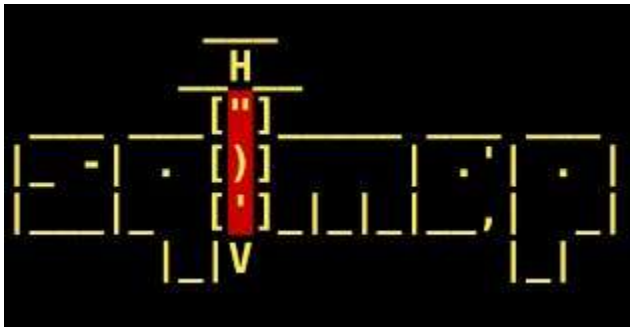
| Category | Indicator | Description | Threat Actor/Tool | Mitigation |
|---|---|---|---|---|
| Protocol | TCP | Protocol used for initiating connection requests to detect open ports. | Nmap | Use network firewalls to limit unnecessary services and filter traffic at the perimeter. |
| Source IP | 192.168.1.9 | IP address of the attacker machine (Kali Linux). | Nmap | Monitor traffic from untrusted sources for unusual activities. |
| Destination IP | 192.168.1.7 | IP address of the target system being scanned. | N/A | Implement intrusion prevention systems (IPS) to detect and block unauthorized scans. |
| Attack Type | Port Scanning | Technique used to identify open ports on the target system. | Nmap | Use rate-limiting, TCP SYN cookies, and fail2ban to block repetitive scan attempts. |
| Data Extraction | Open Ports | Information about available services and their corresponding ports. | Nmap | Configure services to run on non-standard ports and disable unused ports to reduce attack surface. |
| Suricata Alert | "TCP SYN Scan Detected" | Generated alert indicating potential reconnaissance activity. | Suricata IDS | Regularly update IDS rules to include detection for known scanning patterns. |
| Logged Details | Source: 192.168.1.9, Dest: 192.168.1.7, Ports scanned. | Logs captured by Suricata for forensic analysis. | Suricata IDS | Enable centralized log management and periodic review of IDS alerts for suspicious activity. |
| Wireshark Data | TCP SYN packets | Captures packet-level details of the SYN scan, including IP and port information. | Wireshark | Analyze captured traffic to identify scanning patterns and implement blacklisting if necessary. |

## Attack2: Sql Injection

**Overview:**

**Tool Used:**

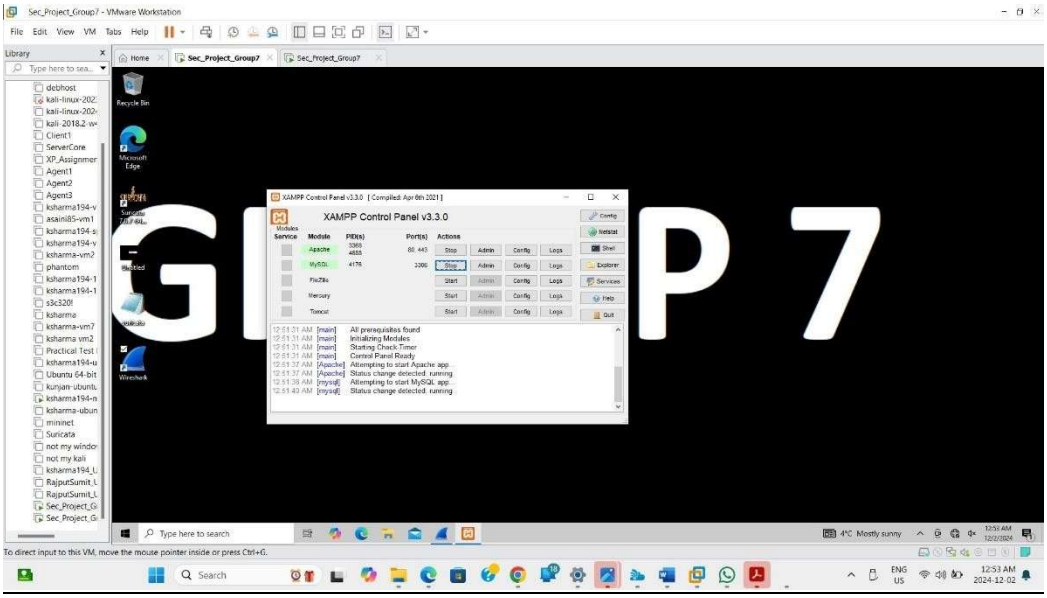
**SQL MAP:**



An open-source penetration testing tool called SQLmap is used to find and take advantage of SQL injection flaws in web applications. By submitting malicious SQL queries to a database, it automates

the process of finding and taking use of these vulnerabilities. Many databases, such as MySQL, PostgreSQL, Oracle, and others, are supported by SQLmap. It can carry out operations such as accessing the underlying server, running arbitrary queries, and retrieving database data.

XAMPP:




The web server solution stack package known as XAMPP is free and open-source. It consists of PHP, Perl, MySQL (or MariaDB), and Apache. Developers frequently use it to install a local server environment on their PCs for testing and development. Web applications may be effectively tested on a local computer before being deployed to a live server thanks to XAMPP's ease of installation and configuration.

DVWA (Damn Vulnerable Web Application):

# SEC320 GROUP 7 PROJECT FINAL REPORT


For the purpose of practicing and teaching web application security to security experts and students, DVWA is a purposefully insecure web application. It is frequently used in schools and penetration testing labs to mimic real-world vulnerabilities including file inclusion, SQL injection, and cross-site scripting (XSS). Through DVWA, users can learn about common web application vulnerabilities and how to take advantage of them in a safe and authorized setting.

Why were these tools used:

**SQLmap**:

SQLmap was created especially to simplify the process of finding and taking advantage of SQL injection flaws in online applications. An attacker can obtain unauthorized access to a database by manipulating a web application's SQL queries through a technique known as SQL injection. SQLmap assists in locating and taking advantage of these weaknesses by:

Identifying SQL injection locations automatically.
Testing various SQL injection attack types (such as time-based, Boolean-based, etc.).
retrieving private information such as tables and user credentials, as well as running arbitrary instructions on the database server.

**XAMPP**:

# SEC320 GROUP 7 PROJECT FINAL REPORT

A popular pairing with SQL injection testing is the local development server software XAMPP. XAMPP offers a secure, isolated environment where developers and security experts may test and experiment with SQL injection techniques without endangering production systems, even if it doesn't explicitly enable SQL injection. Because it runs apps locally, it's perfect for testing security vulnerabilities prior to distribution.

DVWA:

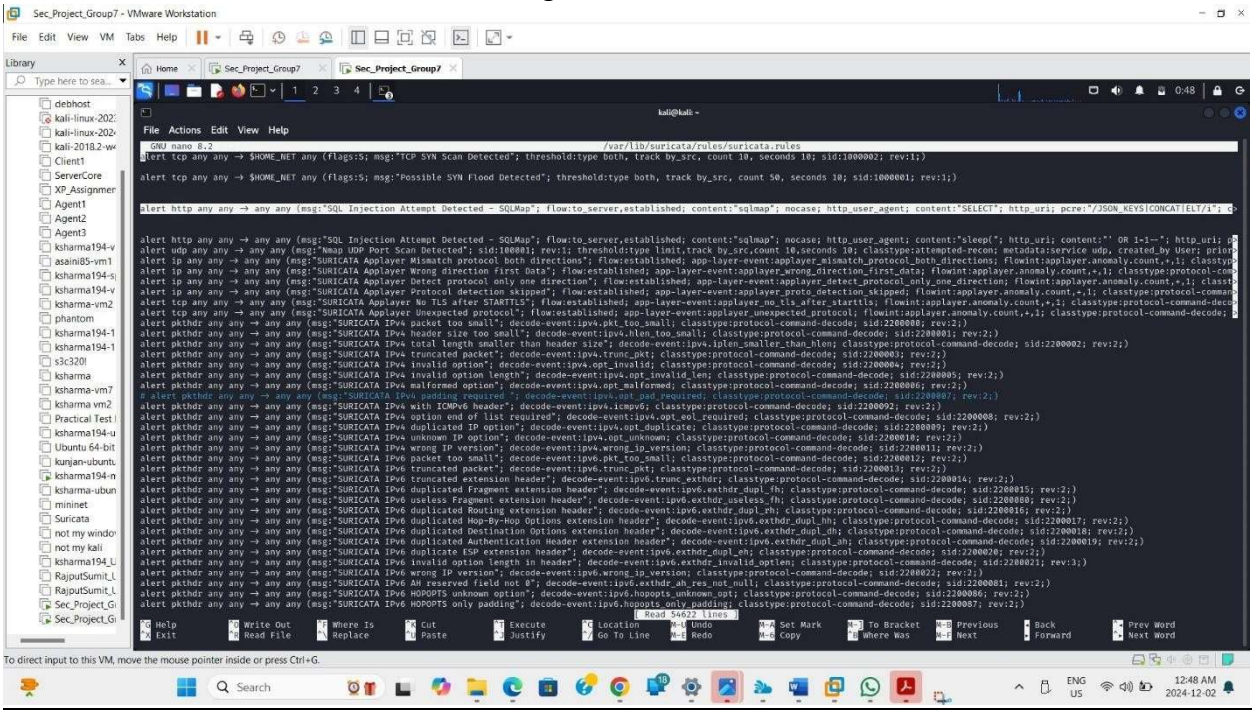
The purpose of DVWA is to create a purposefully weak web application. It is a useful learning tool since it includes intentionally unsafe code that mimics common vulnerabilities, such as SQL injection. In a secure setting, users can practice taking advantage of SQL injection flaws. DVWA is helpful:

Recognize SQL injection and the ways in which attackers take advantage of it.
Evaluate how well SQL injection defences are working.
Learn how to simulate attacks in order to prevent SQL injection vulnerabilities.
When combined, these tools offer a testing environment and the ability to take advantage of SQL injection vulnerabilities, which makes them indispensable for understanding and assessing web application security.

**PREPARATION**:

**File Configuration:**

We configered the file sudo nano /var/lib/suricatta/rules/suricatta. Rules to add the the SQLinjection attack. This allows suricatta to detect and log the attack.

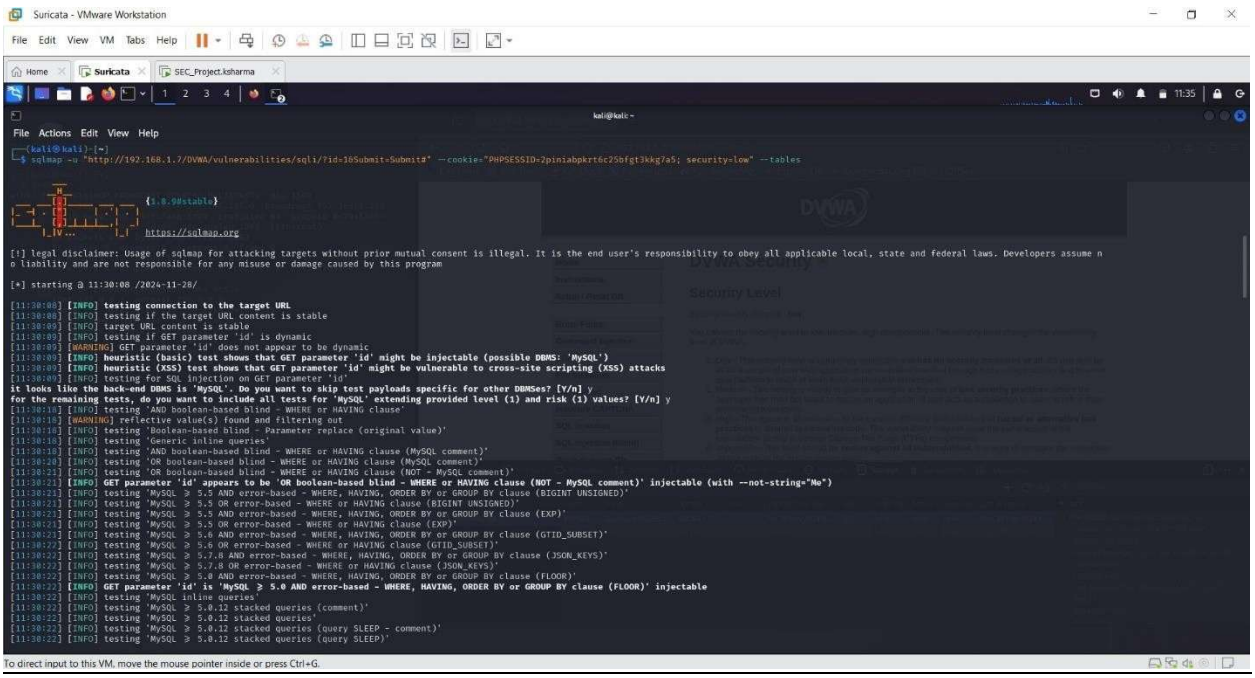


Additionally, we must lower the DVWA security to a low level. By doing this, DVWA will make it less secure and let us observe how a SQL injection attack occurs.

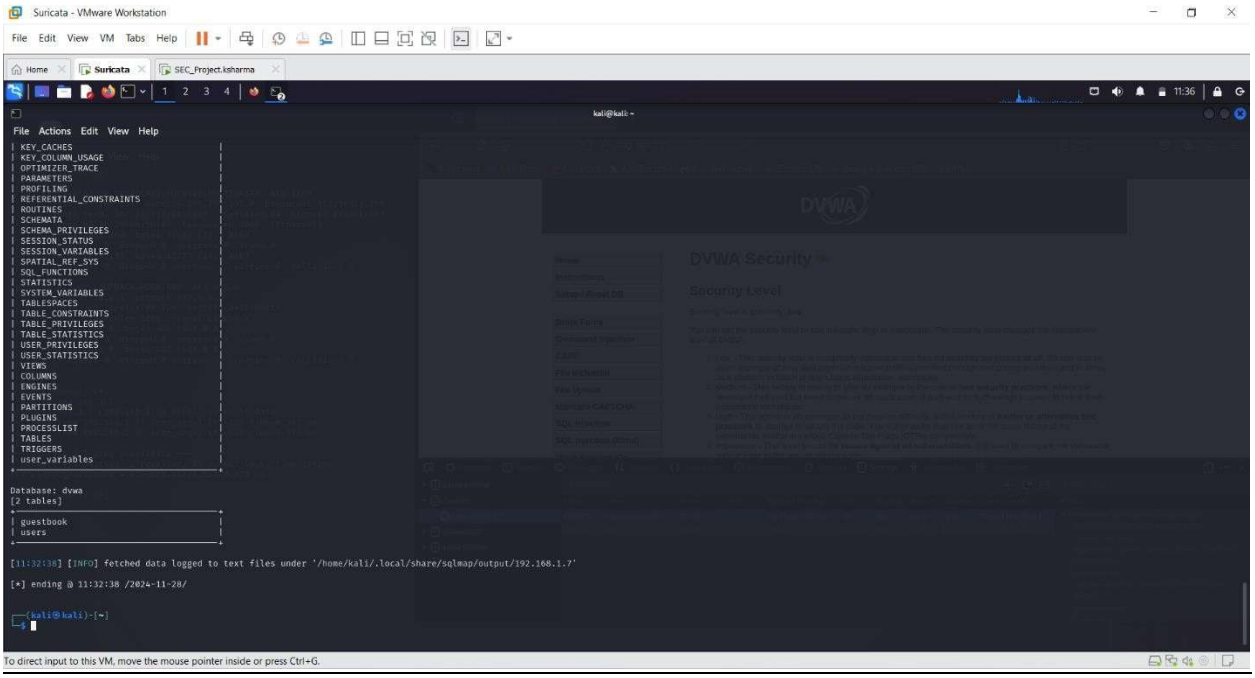
**Attack Execution:**

sqlmap -u "http://192.168.1.7/DVWA/vulnerabilities/sqli/?id=10&Submit=Submit#" -cookie="PHPSESSID=2piniabpkrt6c25bfgt3kkg7a5; security=low" –tables

Target URL: SQLmap makes an HTTP request to the DVWA SQL Injection page's vulnerable id parameter.
SQL Injection Testing: By examining the server's response, SQLmap automatically determines if the id parameter is susceptible to SQL injection.
Session Handling: To guarantee that the vulnerability may be exploited, it sets the DVWA's security level to "low" and supplies a valid session ID (PHPSESSID).
Database Table Enumeration: SQLmap will try to enumerate every table in the DVWA application's database using the --tables option if the id argument is determined to be injectable.



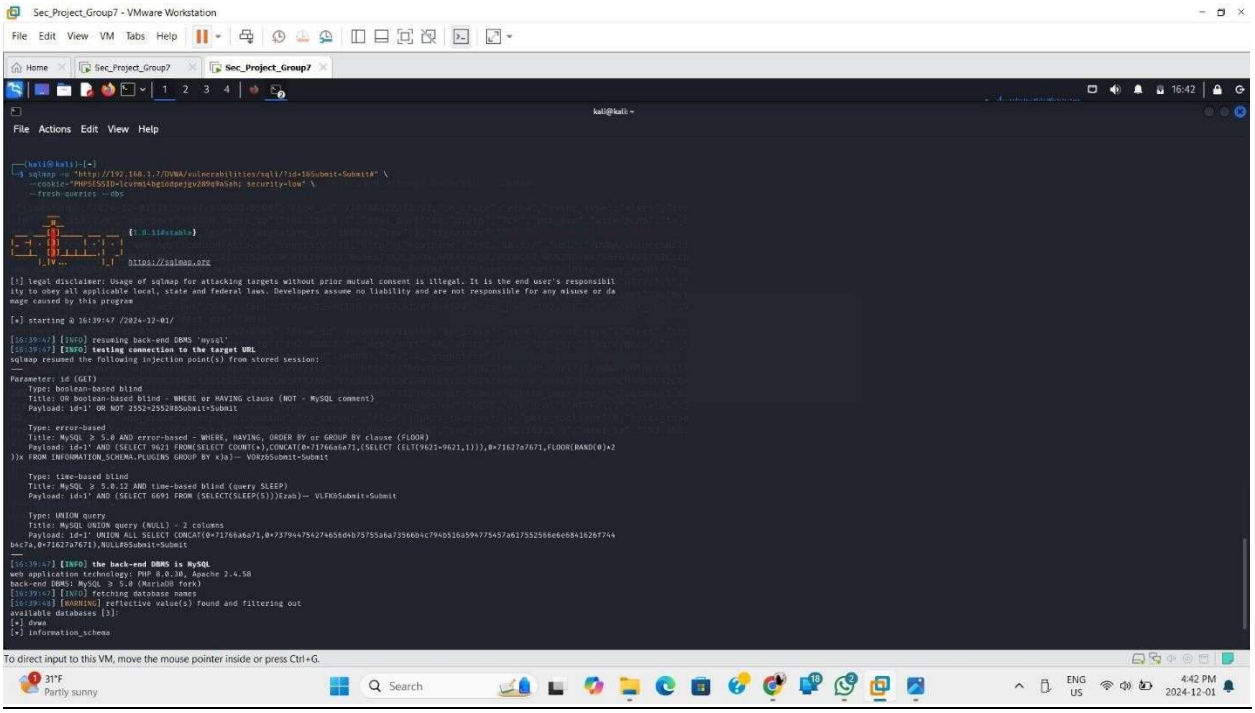
# SEC320 GROUP 7 PROJECT FINAL REPORT




sqlmap -u "https://192.168.1.7/DVWA/vulnerabilities/sqli/?id=10&Submit=Submit#" --cookie="PHPSESSID=vl4bqojidpey92acyt29qw9a5ah; security=low" --fresh-queries –dbs

Target URL: To check for SQL injection vulnerabilities, SQLmap makes an HTTP request to the DVWA SQL Injection page's vulnerable id parameter.
Session Handling: By setting the security level in DVWA to low and including a session cookie (PHPSESSID), the request exposes the application to SQL injection attacks.
New Queries: SQLmap compels the program to run brand-new injection tests and ignore any previously stored SQL queries.
Database Enumeration: SQLmap will utilize the –dbs option to display every database on the target database server if the id parameter is provided.

# SEC320 GROUP 7 PROJECT FINAL REPORT

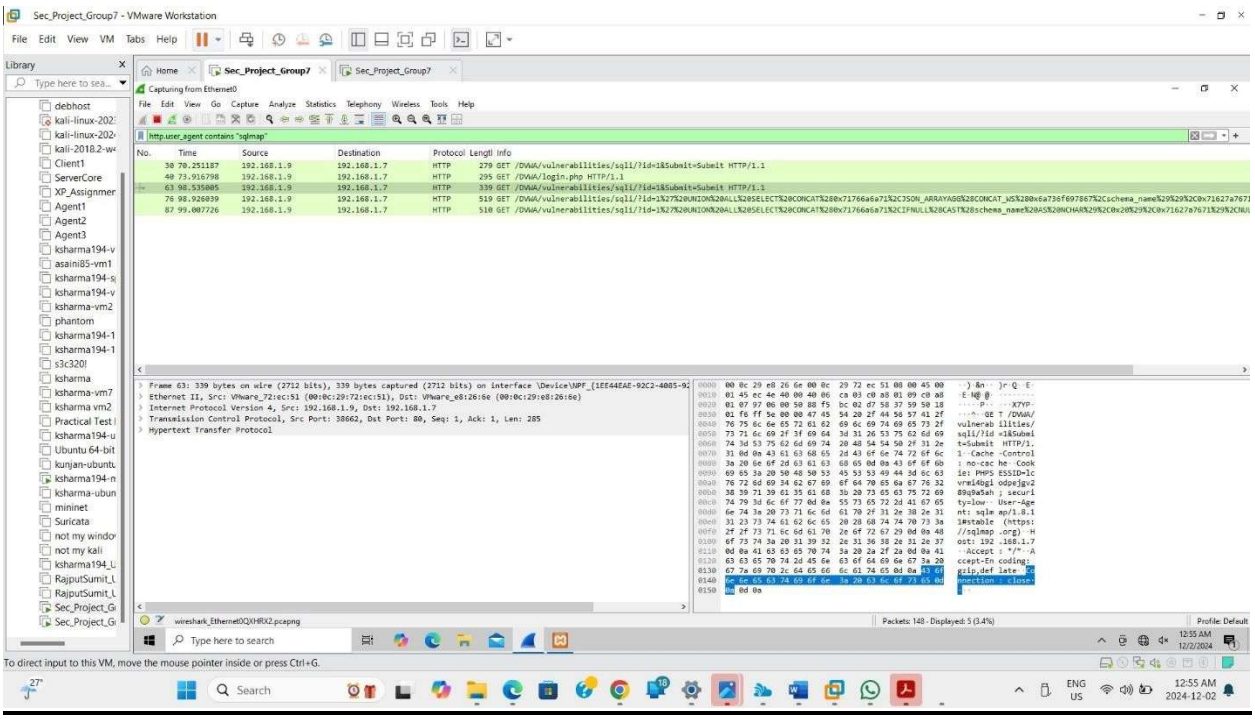



## Observations:

1. **HTTP Requests:**
    - SQLmap sends requests to the target URL, specifically testing the id parameter for vulnerabilities by injecting SQL commands.
    - The requests include session information (PHPSESSID) and a setting for low security (security=low), which makes the application easier to exploit.
2. **SQL Injection Attempts:**
    - You'll notice multiple requests with various SQL injection techniques like ' OR 1=1. These are attempts to exploit the vulnerability in the target system.
    - Responses from the server may contain error messages, such as syntax errors, or normal responses if the injection doesn't work.
3. **Database Information Gathering:**
    - When using the --dbs or --tables options, SQLmap tries to list all databases or tables on the server.
    - If successful, the server will return a list of available databases or tables, which will be visible in Wireshark's response data.
4. **Increased Network Traffic:**

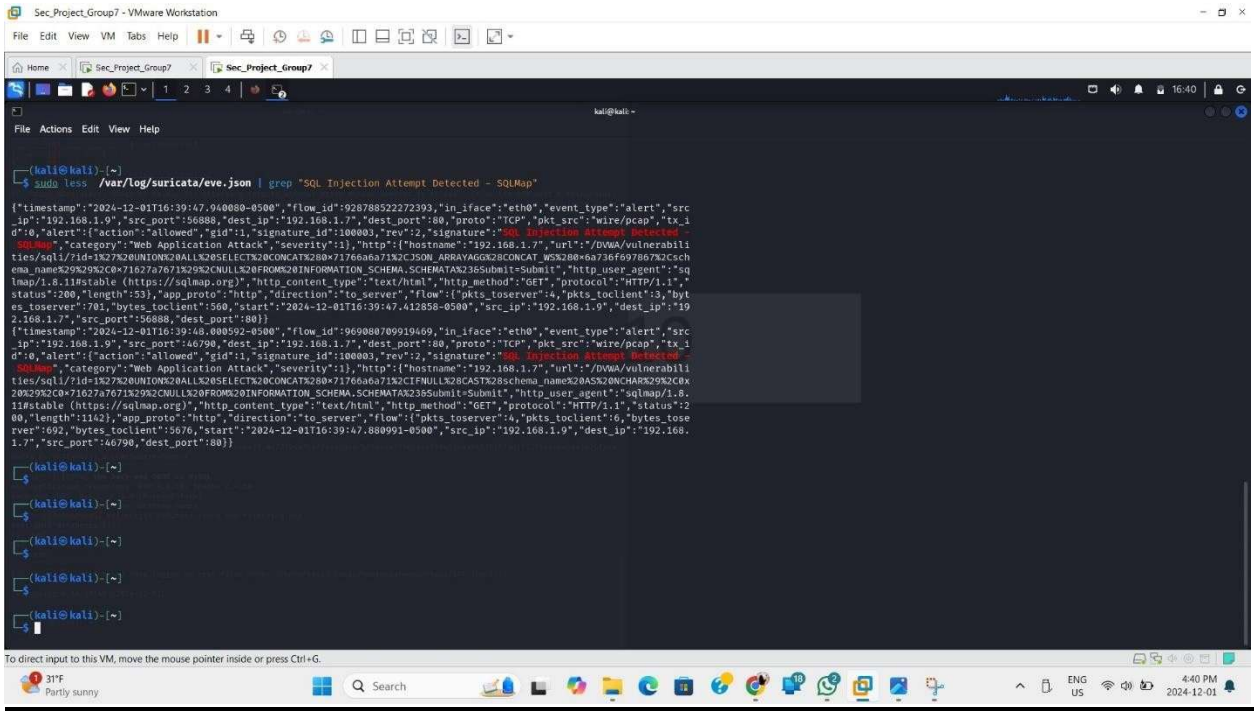
- Because of the aggressive rate set by SQLmap (--min-rate 1000), you'll see a high volume of requests in Wireshark. This indicates that the tool is trying to test the system quickly.
- The fresh queries option ensures that SQLmap doesn't reuse previous results, meaning Wireshark will show repeated requests for new attempts.

In summary, Wireshark will show high-frequency requests with SQL injection payloads and responses that either contain database information or error messages. These patterns are typical of a system being tested for SQL injection vulnerabilities.




**Detection and Evidence:**

# SEC320 GROUP 7 PROJECT FINAL REPORT




## Ways to prevent this attack:

Make use of parameterized queries, or prepared statements:
To stop harmful input from being executed, always utilize prepared statements to keep user input and SQL queries apart.

Validation and Sanitization of Input:
Verify that inputs adhere to the required forms and sanitize them to get rid of dangerous characters like ", ", and --."

Restrict Database Permissions:
To reduce risk, limit database access to only those functions that are required for the web application.

Employ WAFs (Web Application Firewalls):
To screen and stop SQL injection attempts before they reach the server, implement WAFs.

Handling Errors:
Use generic error messages rather than disclosing specific database issues to consumers.
Frequent audits of security:
To find and address vulnerabilities, conduct routine testing and penetration testing.

Make use of HTTPS (secure connections):

# SEC320 GROUP 7 PROJECT FINAL REPORT

To safeguard data integrity and avoid interception, make sure all communications are encrypted using HTTPS.

## IMPACT ON THE VICTIM:

Unauthorized Access and Data Breach:

Attackers may obtain sensitive information stored in the database, such as credit card numbers, usernames, passwords, and personal data, without authorization. Data Corruption or Loss:

Important information may be lost or corrupted as a result of SQL injection, which gives attackers the ability to remove or alter data.
The escalation of privileges

Attackers may increase their privileges and obtain administrator access to the database or system by taking advantage of SQL injection vulnerabilities.
Compromise in the System:

Attackers may be able to install malware, compromise the underlying system, or take over the server by using malicious SQL queries to execute arbitrary commands. Damage to Reputation:

An organization's reputation can be seriously harmed by a successful SQL injection attack, particularly if private client information is lost or compromised.

## Summary Table for the DHCP Spoofing Attack:

| Attack Name | Launched (formula) | Indicator1 | Indicator2 | Indicator3 | Indicator4 | Possible Tools |
|---|---|---|---|---|---|---|
| SQL INJECTION | Launched using user input parameters in web applications | Unusual database error messages | Unauthorized data access or retrieval | Unexpected database changes (e.g., deleted or | Highfrequency HTTP requests (with SQL | SQLmap, DVWA, XAMPP |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | modified data) | injection payloads) | |

## Indicator Conversion table:

| Category | Indicator | Description | Threat Actor/Tool | Mitigation |
|---|---|---|---|---|
| Protocol | HTTP | Protocol used for sending web requests to interact with the vulnerable website. | SQLMap | Implement input validation and parameterized queries to sanitize user input. |
| Source IP | 192.168.1.9 | IP address of the attacker machine (Kali Linux). | SQLMap | Monitor traffic from external sources for abnormal activities. |
| Destination IP | 192.168.1.7 | IP address of the target system hosting DVWA (Damn Vulnerable Web Application). | N/A | Apply access controls to limit access to the application only to trusted systems. |
| Attack Type | SQL Injection | Code injection technique to manipulate database queries through vulnerable input fields. | SQLMap | Use Web Application Firewalls (WAF) to detect and block SQL injection patterns. |
| Data Extraction | Database tables and credentials | SQLMap extracts data such as table names, user credentials, and other sensitive data from the database. | SQLMap | Encrypt sensitive data in the database and limit database user privileges. |
| Suricata Alert | "SQL Injection Attempt Detected" | Generated alert indicating malicious SQL injection attempts. | Suricata IDS | Update IDS rules regularly to include the latest SQL injection patterns. |
| Logged Details | Source: 192.168.1.9, Dest: 192.168.1.7, Queries | Logs captured by Suricata for forensic analysis. | Suricata IDS | Enable log monitoring and perform regular reviews of logs. |
| Wireshark Data | HTTP requests and responses | Captures packet-level details of SQL injection attempts on the DVWA server. | Wireshark | Analyze captured network traffic for unauthorized database queries. |