As an **SDET (Software Development Engineer in Test)**, learning **Kali Linux** can help you enhance your security testing skills, penetration testing, and vulnerability assessment in test automation. Here's what you should focus on:

---

1. Linux Basics & Shell Scripting

- Basic Linux commands (`ls`, `cd`, `cp`, `mv`, `rm`, `chmod`, `grep`, etc.)
- File and process management (`ps`, `top`, `kill`, `crontab`, etc.)
- Shell scripting (`bash`, `awk`, `sed`, `cron jobs` for automation)

---

2. Networking & Security Testing

- **Networking Basics**: `ifconfig`, `ip`, `ping`, `traceroute`, `netstat`, `nmap`
- **Port Scanning**: `nmap` for open port detection
- **Packet Sniffing**: `Wireshark`, `tcpdump`
- **Man-in-the-Middle Attacks**: `ettercap`, `MITMf`
- **Intercepting HTTP Requests**: `Burp Suite`, `ZAP Proxy`

---

3. Penetration Testing & Ethical Hacking

- **Metasploit Framework**: Exploiting web applications, databases, and networks
- **SQL Injection Testing**: `sqlmap`
- **Password Cracking**: `John the Ripper`, `Hydra`, `Hashcat`
- **Web Application Security Testing**:

    - `OWASP ZAP` for finding vulnerabilities
    - `Nikto` for scanning web servers
- **File Forensics**: `Autopsy`, `foremost`
- **Reverse Engineering**: `Ghidra`, `Radare2`

---

4. API Security Testing

- **Using Postman for API testing**
- **Automating API security testing** with `OWASP ZAP` and `Burp Suite`
- **Fuzz Testing**: `ffuf`, `wfuzz`
- **Automating API vulnerability scanning** with `Nikto` and `sqlmap`

---

5. Mobile Security Testing

- **Testing Android apps** with `MobSF` (Mobile Security Framework)
- **APK Decompilation**: `apktool`, `dex2jar`
- **Traffic Analysis**: `mitmproxy`, `Burp Suite`
- **Checking for insecure storage**: `drozer`

---

6. Scripting for Automation & Security Testing

- Writing **Python scripts** for security testing (`requests`, `BeautifulSoup`, `Selenium`, `Scapy`)

- Automating **penetration tests** with Python & Bash scripting

---

7. Log Analysis & SIEM

- **Log parsing** using `grep`, `awk`, `sed`
- **Analyzing system logs** (`/var/log/syslog`, `auth.log`, `dmesg`)
- **Using ELK Stack (Elasticsearch, Logstash, Kibana)** for log analysis

---

8. Performance & Load Testing (Security Focused)

- **Testing security under load** using `JMeter` and `Gatling`
- **Identifying security issues in high-traffic scenarios**

---

9. CI/CD & Security Integration

- Integrating **security testing tools** (`ZAP`, `Burp Suite`, `OWASP Dependency Check`) into **Jenkins/GitHub Actions**
- Running **automated security scans** as part of CI/CD pipelines

---

10. Learning About Cyber Threats & Attack Vectors

- **Malware Analysis**: `ClamAV`, `YARA`
- **Phishing Attacks & Prevention**
- **Exploiting and fixing vulnerabilities** in Web Applications

---

How Kali Linux Helps in SDET Role

- **Security Testing** for web & mobile applications
- **API Penetration Testing** for secure API development
- **Automation of security checks** in test scripts
- **Vulnerability scanning & remediation** in SDLC

Next Steps:
Since you are already learning **Metasploit in Kali Linux**, start integrating security testing into your **SDET workflow** by automating scans and API security tests. Would you like practical exercises or project ideas to apply this knowledge? 🚀

4o

You said:

yes

ChatGPT said:

Here are **practical exercises and project ideas** to apply **Kali Linux skills** for **SDET (Security Testing & Automation)**:

🔹 Practical Exercises (Step-by-Step)

1️⃣ Automate Security Scanning in API Testing

Goal: Identify security vulnerabilities in APIs during automation testing.

✅ **Steps**:

1. Install **OWASP ZAP** in Kali (`apt install zaproxy`)
2. Write a **Python script** to automate API security scans using ZAP's API

3. Run a **POST request attack simulation** (e.g., SQL Injection, XSS)
4. Capture vulnerabilities in a report
5. Integrate this script into **Cypress** or **Selenium test cases**

Tools: OWASP ZAP, Python, Postman, Cypress

🚀 **Outcome**: Automate API security scans in CI/CD pipelines.

2️⃣ Automate SQL Injection & Security Testing

Goal: Detect SQL Injection vulnerabilities in a login page.

✅ **Steps**:

1. Find a vulnerable login page (use **DVWA** or **bWAPP** in Kali)
2. Use `sqlmap` to check for SQL Injection (`sqlmap -u "http://target.com/login.php?user=admin" --dbs`)
3. Write a **Python script** to automate **SQL Injection Testing**
4. Add **assertions** in Cypress or Selenium tests to flag security risks
5. Generate an **HTML security report** after execution

Tools: SQLMap, DVWA, Selenium, Cypress, Python

🚀 **Outcome**: Automated SQL Injection detection in test automation.

3️⃣ Security Test a Web App with Burp Suite

Goal: Test and intercept security vulnerabilities in a website.

✅ **Steps**:

1. Open **Burp Suite** in Kali

2. Configure **Firefox proxy settings** to route traffic via Burp
3. Capture **requests and responses** from a test web app
4. Perform **man-in-the-middle attacks** (e.g., change HTTP responses)
5. Automate the testing using **Burp Suite API** and **Python scripts**

Tools: Burp Suite, Python, Selenium

🚀 **Outcome**: Automate Burp Suite in security test cases.

4️⃣ Automate Vulnerability Scanning in CI/CD

Goal: Integrate security scanning into Jenkins or GitHub Actions.

✅ **Steps**:

1. Install **Nikto** in Kali (`apt install nikto`)
2. Run a **web security scan** (`nikto -h http://target.com`)
3. Write a **Bash script** to trigger Nikto scans in CI/CD
4. Store results in **Jenkins logs** for security analysis
5. Add a **fail condition** if critical vulnerabilities are found

Tools: Nikto, Bash, Jenkins, GitHub Actions

🚀 **Outcome**: Security checks become part of the CI/CD process.

5️⃣ API Penetration Testing with Metasploit

Goal: Exploit a vulnerable API using Metasploit.

✅ **Steps**:

1. Find an **exploitable API** (e.g., Open APIs on **Shodan**)

2. Use Metasploit (`msfconsole`) to scan for API weaknesses
3. Try running an **authentication bypass exploit**
4. Capture logs using **Wireshark**
5. Write a **Python automation script** to monitor API security

Tools: Metasploit, Python, Wireshark, OWASP ZAP

🚀 **Outcome**: API security tests can be automated in regression testing.

🔹 Project Ideas (Real-World Security Testing)

1️⃣ Secure Web Testing Automation Suite

- Automate **SQL Injection**, **XSS**, and **CSRF** tests in **Selenium & Cypress**
- Use **ZAP Proxy API** to scan websites dynamically
- Generate **security reports** in CI/CD pipelines

Tech Stack: Cypress, Selenium, Python, OWASP ZAP

2️⃣ Security Testing for Mobile Apps

- Use **MobSF** to analyze Android APK security
- Perform **reverse engineering** using `apktool`
- Run automated **fuzz tests** on mobile API endpoints

Tech Stack: MobSF, APKTool, Python, Bash

3️⃣ Automated API Security Monitoring System

- Monitor **API traffic** using **Wireshark**
- Automate **intrusion detection** using **Metasploit & Python**
- Send alerts to **Slack/Email** when vulnerabilities are found

Tech Stack: Metasploit, Wireshark, Python

4️⃣ Real-Time Penetration Testing Dashboard

- Create a **dashboard** that scans for security vulnerabilities daily
- Display **SQL Injection, XSS, and authentication risks**
- Use **Grafana** for real-time security monitoring

Tech Stack: Python, OWASP ZAP, Grafana, Bash

🚀 Next Steps

Which project or exercise do you want to start with? I can help with step-by-step implementation! 😊

-