Requirement 11: Regularly Test Security Systems and Processes

This requirement ensures regular testing of security systems to detect and address vulnerabilities, covering internal and external threats.

11.1: Test and Detect Authorized and Unauthorized Access Points (APs)

- **Details**: Regular scans should be performed to detect rogue access points (APs). Incident response procedures must also be in place.
- **Laravel Implementation**:
  - Use Laravel tasks (e.g., **Laravel Scheduler**) to schedule scans of the network regularly.
  - Create a dashboard in Laravel Nova to monitor and alert administrators of rogue APs.
- **Third-Party Tools**:
  - **Kismet** for wireless network detection.
  - **Wireshark** for network protocol scanning.
  - **Cisco Meraki** for AP monitoring and alerts.

11.2: Internal and External Vulnerability Scans

- **Details**: Vulnerability scans must be done quarterly and after major changes. They should be performed by an ASV (Approved Scan Vendor) with a passing score of 3.9 or lower on the CVSS scale.
- **Laravel Implementation**:
  - Integrate Laravel with a vulnerability scanning API that tracks and reports on the health of the application.
  - Automate vulnerability scans using Laravel queues and jobs.
- **Third-Party APIs**:
  - **Tenable.io** or **Qualys** for vulnerability scanning.
  - **AWS Inspector** for application-level vulnerability checks.

11.3: Internal and External Penetration Testing

- **Details**: Penetration testing should be conducted at least once a year and after significant changes, following a documented methodology (e.g., NIST SP800-115*). It should cover network, application, and protocol layers.
- **Laravel Implementation**:
  - Schedule annual penetration tests using Laravel queues or cron jobs.
  - Develop a module in Laravel Nova to store and review penetration test results and improvements.
- **Third-Party APIs**:
  - **Burp Suite**, **OWASP ZAP**, or **Metasploit** for penetration testing.
  - **Pentest-Tools.com** for automated penetration testing.

11.4: Use of Intrusion Detection Systems (IDS) or Intrusion Prevention Systems (IPS)

- **Details**: IDS or IPS software should monitor critical locations (e.g., database servers). They must be kept up to date.

- **Laravel Implementation**:
  - Integrate an IDS/IPS monitoring tool with Laravel for logging and detecting potential threats.
  - Use Laravel's notification system to alert administrators of any detected intrusions.
- **Third-Party APIs**:
  - **Suricata** or **Snort** for intrusion detection and prevention.
  - **CrowdStrike** or **AlienVault** for managed IDS/IPS solutions.

11.5: File Integrity Monitoring (FIM) and Change Detection

- **Details**: Regular file integrity monitoring should check critical files (system files, logs, configuration files) at least once a week. Alerts should be in place for unauthorized changes.
- **Laravel Implementation**:
  - Use Laravel’s task scheduling to run FIM checks on critical files weekly.
  - Create a logging system in Laravel that captures file changes and sends alerts when unauthorized changes occur.
- **Third-Party APIs**:
  - **Tripwire** for file integrity monitoring.
  - **Wazuh** for real-time file monitoring and change detection.

11.6: Document and Enforce Policies and Procedures

- **Details**: All security testing procedures, schedules, and actions must be properly documented and enforced.
- **Laravel Implementation**:
  - Use Laravel’s **Eloquent ORM**** to store and manage security policies and procedures in a database.
  - Implement a workflow system using **Laravel Nova** to track and enforce the procedures, ensuring compliance.
- **Third-Party APIs**:
  - **Confluence** or **Notion** for documentation management.
  - **Jira** for tracking and enforcing security-related tasks and procedures.

Key Takeaways:

- Regular testing and monitoring, from vulnerability scans to penetration testing, are essential for maintaining a secure environment.
- Laravel's powerful scheduling and notification systems can help automate and manage many of these testing procedures.
- Third-party APIs such as **Qualys**, **Burp Suite**, **Tenable.io**, and **Snort** offer robust security testing tools that can be integrated into Laravel applications for enhanced monitoring.

These implementations ensure that security systems and processes are tested regularly to detect vulnerabilities and maintain compliance with security standards like PCI DSS.

**Instead of storing the policies and procedures directly in a database, you can:

- **Store links to documents** (e.g., in Confluence, SharePoint) in the database using Eloquent if needed, for easy access.

- Use **document management tools** (e.g., Confluence, Notion, or a Git repository) to manage security documents, which provide version control and better access control.
- Use **Laravel Nova** for tracking compliance-related tasks and enforcing procedures through workflows (e.g., ensuring all employees have read and signed off on a particular security policy).

*NIST SP800-115, titled **"Technical Guide to Information Security Testing and Assessment,"** is a guide published by the National Institute of Standards and Technology (NIST) that provides organizations with technical methods for testing and assessing information security systems. It is designed to help ensure the confidentiality, integrity, and availability of systems by outlining various techniques and approaches for conducting security assessments.

Key Aspects of NIST SP800-115

1. **Purpose**

    - The guide provides organizations with a methodology for assessing the effectiveness of their security controls through technical information security tests. It focuses on identifying vulnerabilities, weaknesses, and potential security issues that could be exploited by attackers.

2. **Testing and Assessment Methods** NIST SP800-115 divides information security testing into three primary categories:

    - **Examination**: A process of reviewing, inspecting, and analyzing security-related information (e.g., policies, configuration settings, or documentation).
        - Examples: Reviewing security policies, configuration settings, firewall rules, etc.
    - **Interviews**: A process of gathering information from personnel to understand their responsibilities, knowledge, and activities regarding security measures.
        - Examples: Speaking with system administrators, security officers, or developers.
    - **Testing**: The process of evaluating the security mechanisms in place, including exploiting weaknesses to determine how systems react under attack.
        - Examples: Penetration testing, vulnerability scanning, social engineering, etc.

3. **Phases of Security Testing** The guide divides the security assessment process into four distinct phases:

    - **Planning**: The phase where goals and objectives for the assessment are defined. This includes understanding the systems, boundaries, and scope of the assessment, selecting appropriate testing methods, and setting up rules of engagement.
    - **Discovery**: The process of gathering information about the target environment to identify security weaknesses and vulnerabilities. This is typically where scanning, fingerprinting, and enumeration techniques are used.
    - **Attack/Exploitation**: This phase involves actively trying to exploit identified vulnerabilities to determine whether they can be leveraged to compromise system security.

    - **Post-Attack/Reporting**: After testing is complete, this phase involves reporting the findings, evaluating the results, and recommending remediations or mitigation strategies to address any identified issues.

4. **Types of Security Testing** The guide outlines various types of testing that can be employed during the assessment:
    - **Network Testing**: Identifying vulnerabilities in network infrastructure, services, protocols, etc.
        - Techniques: Port scanning, vulnerability scanning, network mapping, etc.
    - **Application Security Testing**: Examining the security of web or client-server applications.
        - Techniques: Input validation testing, authentication, and authorization assessments.
    - **Physical Security Testing**: Assessing the physical security controls such as access to secure areas, data centers, and equipment.
    - **Wireless Testing**: Assessing wireless network security, including unauthorized access points and encryption weaknesses.
    - **Social Engineering**: Testing the human element of security by using techniques like phishing, impersonation, or tailgating to exploit personnel vulnerabilities.
5. **Ethical Considerations**
    - NIST SP800-115 emphasizes the importance of ethics and compliance when conducting security assessments. The testing should always be authorized, with clearly defined scope, objectives, and rules of engagement to avoid violating privacy or causing unintended disruptions.
6. **Reporting**
    - **Document Findings**: Once testing is completed, the results should be documented in a comprehensive report. The report should include all identified vulnerabilities, their potential impacts, the exploited vulnerabilities during testing, and any remediation recommendations.
    - **Recommendations**: Providing actionable suggestions for improving the security posture of the systems tested, including patching, system configuration improvements, and policy updates.
7. **Benefits of NIST SP800-115**
    - Helps organizations identify weaknesses before attackers can exploit them.
    - Ensures systems and networks are properly configured and secure.
    - Assists in meeting compliance requirements for various security standards, including PCI DSS, HIPAA, and others.

Summary of Key Techniques in NIST SP800-115

1. **Vulnerability Scanning**: Automated tools used to identify security weaknesses, such as unpatched systems or weak configurations.
2. **Penetration Testing**: Simulating attacks on systems to find vulnerabilities by exploiting them.
3. **Security Control Assessments**: Verifying the implementation and effectiveness of security controls.

4. **Wireless Security Testing**: Assessing the security of wireless networks by identifying rogue access points or weak encryption.
5. **Social Engineering**: Testing human weaknesses through techniques like phishing or impersonation.
6. **Configuration Reviews**: Verifying security settings on systems, networks, and devices.

Conclusion

NIST SP800-115 provides a comprehensive framework for performing information security assessments in an organized and ethical manner. It guides organizations through the phases of testing, ensures the proper selection of testing methods, and provides a structured approach for identifying and mitigating security weaknesses.