Security Architecture and Frameworks Summative Assessment Report

Name: [Alexander Focsha]
Date: [06/06/2025]
Role: Junior Cybersecurity Engineer

Executive Summary

This report details the assessment of the organization's IT infrastructure against PCI DSS compliance standards, with a focus on network segmentation, vulnerability management, and secure data transfer. The assessment involved configuring VyOS firewall rules, hardening web and database servers, analyzing existing automated data transfer processes, and conducting security scans using Wazuh and Nessus.

Key findings indicate that while robust network segmentation is in place, the web server presented vulnerabilities to SQL injection and XSS. The automated data transfer process, while secure, could benefit from alternative strategies for improved efficiency. Recommendations are provided to enhance security posture and ensure ongoing PCI DSS compliance.

Network Segmentation Enforcement

Purpose of Network Segmentation

Network segmentation is a critical security control for organizations handling sensitive data, especially those subject to PCI DSS. Its primary purpose is to isolate systems, reducing the scope of a potential breach and limiting lateral movement for attackers. By dividing the network into smaller, isolated segments, an organization can:

- **Reduce the attack surface:** Limiting direct communication between various network zones minimizes exposure to threats.
- **Contain breaches:** If one segment is compromised, the impact is localized, preventing the breach from spreading to other critical systems, particularly those storing PCI data.
- **Enforce granular access control:** Specific firewall rules can be applied to each segment, dictating precisely which traffic is allowed or denied.

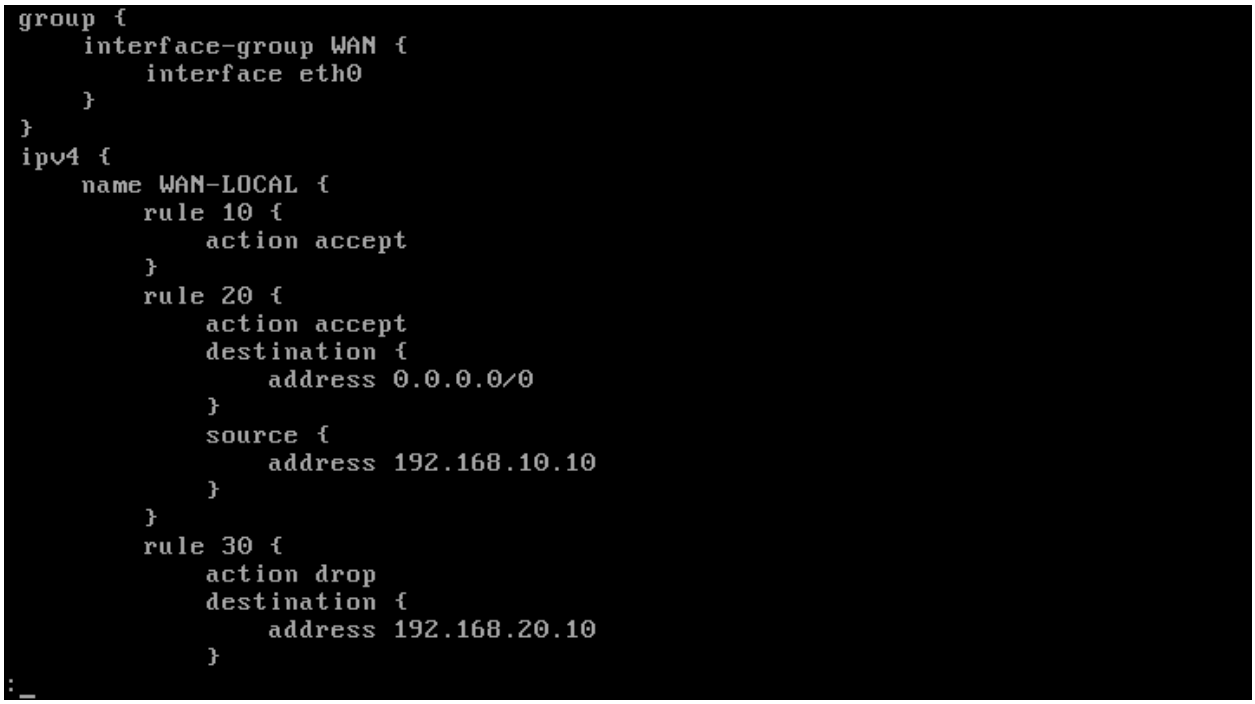
- **Simplify compliance:** By clearly defining the boundaries of the Cardholder Data Environment (CDE), it becomes easier to apply and audit PCI DSS controls.

In this assessment, a strategic segmentation approach has been implemented to isolate the Web Server, Database Server, and the (temporarily offline) Internal Application Server from each other and from external access, while allowing necessary communication with the Compliance Server for monitoring and secure data transfers.

Firewall Configuration and Implementation

The VyOS router is configured with firewall rules to enforce strict network segmentation. The `WAN-LOCAL` firewall name is applied, controlling traffic originating from interfaces connected to the internal network segments and destined for the router itself, or for other internal segments, after passing through the router.

Screenshot of VyOS Firewall Rules:



```
            action drop
            destination {
                address 192.168.20.10
            }
            source {
                address 192.168.10.10
            }
        }
        rule 40 {
            action drop
            destination {
                address 192.168.30.10
            }
            source {
                address 192.168.10.10
            }
        }
        rule 50 {
            action drop
            destination {
                address 192.168.20.10
            }
            source {
                address 0.0.0.0/0
:_
```

```
        rule 50 {
            action drop
            destination {
                address 192.168.20.10
            }
            source {
                address 0.0.0.0/0
            }
        }
        rule 60 {
            action drop
            destination {
                address 192.168.30.10
            }
            source {
                address 0.0.0.0/0
            }
        }
        rule 70 {
            action accept
            destination {
                address 192.168.40.10
            }
            source {
:
```

```
        rule 70 {
            action accept
            destination {
                address 192.168.40.10
            }
            source {
                address 192.168.10.10
            }
        }
        rule 80 {
            action accept
            destination {
                address 192.168.40.10
            }
            source {
                address 192.168.20.10
            }
        }
        rule 90 {
            action accept
            destination {
                address 192.168.40.10
            }
            source {
:
```

```
        rule 80 {
            action accept
            destination {
                address 192.168.40.10
            }
            source {
                address 192.168.20.10
            }
        }
        rule 90 {
            action accept
            destination {
                address 192.168.40.10
            }
            source {
                address 192.168.30.10
            }
        }
        rule 100 {
            action drop
        }
    }
}
[edit]
vyos@vyos# _
```

Description of Firewall Rules and Their Necessity:

The following rules have been implemented to achieve the required segmentation:

- **Rule 10: Allow Established Connections**
    - `set firewall ipv4 name WAN-LOCAL rule 10 action 'accept'`
    - `set firewall ipv4 name WAN-LOCAL rule 10 state established 'enable'`
    - **Necessity:** This rule is crucial for allowing return traffic for legitimate, established connections. It ensures that once a connection is initiated and accepted, the subsequent packets belonging to that connection are permitted, maintaining operational functionality. This aligns with the principle of least privilege by only allowing traffic that is part of an already approved communication.
- **Rule 20: Allow Web Server External Communication (Outbound)**
    - `set firewall ipv4 name WAN-LOCAL rule 20 action 'accept'`
    - `set firewall ipv4 name WAN-LOCAL rule 20 source address '192.168.10.10'`
    - `set firewall ipv4 name WAN-LOCAL rule 20 destination address '0.0.0.0/0'`
    - **Necessity:** This rule specifically allows the Web Server (192.168.10.10) to initiate outbound connections to external networks (represented by `0.0.0.0/0`). This is necessary for the web server to serve customer-facing applications and potentially fetch updates or external resources. PCI DSS requires limiting outbound connections to only those that are business-justified.
- **Rule 30: Drop Web Server to Database Server Traffic**
    - `set firewall ipv4 name WAN-LOCAL rule 30 action 'drop'`
    - `set firewall ipv4 name WAN-LOCAL rule 30 source address '192.168.10.10'`
    - `set firewall ipv4 name WAN-LOCAL rule 30 destination address '192.168.20.10'`
    - **Necessity:** This rule explicitly denies direct communication between the Web Server and the Database Server. This is a critical PCI DSS requirement (Requirement 1.2.1, 1.3) to isolate the CDE. By preventing direct access, even if the web server is compromised, an attacker cannot directly access the sensitive data on the database server.

- **Rule 40: Drop Web Server to Internal Server Traffic**
    - `set firewall ipv4 name WAN-LOCAL rule 40 action 'drop'`
    - `set firewall ipv4 name WAN-LOCAL rule 40 source address '192.168.10.10'`
    - `set firewall ipv4 name WAN-LOCAL rule 40 destination address '192.168.30.10'`
    - **Necessity:** Similar to Rule 30, this rule isolates the Web Server from the Internal Application Server. Given the Internal Server is currently under forensic investigation, this isolation is paramount to prevent any further compromise or interference with the investigation.
- **Rule 50: Drop All Traffic to Database Server (Except from Compliance Server)**
    - `set firewall ipv4 name WAN-LOCAL rule 50 action 'drop'`
    - `set firewall ipv4 name WAN-LOCAL rule 50 source address '0.0.0.0/0'`
    - `set firewall ipv4 name WAN-LOCAL rule 50 destination address '192.168.20.10'`
    - **Necessity:** This rule is a broad denial that blocks all traffic to the Database Server from any source. This ensures the Database Server remains isolated from external access and all other internal segments, except for explicitly allowed communication from the Compliance Server (covered by a later rule). This strongly aligns with PCI DSS Requirement 1.2, which mandates isolating the CDE from out-of-scope networks.
- **Rule 60: Drop All Traffic to Internal Server (Except from Compliance Server)**
    - `set firewall ipv4 name WAN-LOCAL rule 60 action 'drop'`
    - `set firewall ipv4 name WAN-LOCAL rule 60 source address '0.0.0.0/0'`
    - `set firewall ipv4 name WAN-LOCAL rule 60 destination address '192.168.30.10'`
    - **Necessity:** Similar to Rule 50, this rule isolates the Internal Application Server from all external and other internal network traffic. This is crucial as the server is compromised and under investigation, preventing any unauthorized access or data exfiltration.
- **Rule 70: Allow Web Server to Compliance Server Traffic**

    - `set firewall ipv4 name WAN-LOCAL rule 70 action 'accept'`
    - `set firewall ipv4 name WAN-LOCAL rule 70 source address '192.168.10.10'`
    - `set firewall ipv4 name WAN-LOCAL rule 70 destination address '192.168.40.10'`
    - **Necessity:** This rule permits the Web Server to communicate with the Compliance Server. This communication is essential for security monitoring (Wazuh agent communication) and for the secure transfer of database backups as part of the automated process.
- **Rule 80: Allow Database Server to Compliance Server Traffic**
    - `set firewall ipv4 name WAN-LOCAL rule 80 action 'accept'`
    - `set firewall ipv4 name WAN-LOCAL rule 80 source address '192.168.20.10'`
    - `set firewall ipv4 name WAN-LOCAL rule 80 destination address '192.168.40.10'`
    - **Necessity:** This rule allows the Database Server to communicate with the Compliance Server. This is necessary for the Compliance Server to push data to the Database Server (e.g., forwarded backups) and for security monitoring.
- **Rule 90: Allow Internal Server to Compliance Server Traffic**
    - `set firewall ipv4 name WAN-LOCAL rule 90 action 'accept'`
    - `set firewall ipv4 name WAN-LOCAL rule 90 source address '192.168.30.10'`
    - `set firewall ipv4 name WAN-LOCAL rule 90 destination address '192.168.40.10'`
    - **Necessity:** This rule maintains the connectivity for the Internal Application Server to the Compliance Server. Even while offline, this ensures that once it is brought back online, it can immediately resume security monitoring and potentially participate in data transfers via the Compliance Server, aligning with its intended network role post-investigation.
- **Rule 100: Drop All Other Traffic**

    - `set firewall ipv4 name WAN-LOCAL rule 100 action 'drop'`
    - **Necessity:** This is an explicit "deny all" rule at the end of the firewall policy. It ensures that any traffic not explicitly permitted by the preceding rules is dropped, implementing a strong default-deny posture. This is a fundamental security best practice and a key component of PCI DSS Requirement 1, enforcing strict control over network traffic.

Justification for Rule Choices:

These firewall rules are meticulously designed to protect PCI-sensitive assets while maintaining essential operational functionality. They adhere to the principle of least privilege, allowing only necessary communication paths.

- **PCI DSS Alignment:**
    - **Requirement 1.2.1:** Building firewall and router configurations that restrict connections between untrusted networks and any system components in the CDE. Rules 30, 40, 50, and 60 directly address this by preventing direct communication to the Database Server and Internal Server from unauthorized segments.
    - **Requirement 1.3:** Restricting inbound and outbound traffic to that which is necessary for the CDE. Rules 20, 70, 80, and 90 define the necessary communication, while rules 30, 40, 50, 60, and 100 enforce the restrictions.
    - **Requirement 1.4:** Installing firewalls between all trusted and untrusted networks. The VyOS router acts as this firewall, segmenting the network into trusted (internal) and less trusted (external) zones.
    - **Requirement 2.2:** Developing configuration standards for all system components. The consistent application of these rules across the network demonstrates a structured approach to security.

The segmentation ensures that the Web Server, which is customer-facing, cannot directly access the Database Server, thus protecting sensitive cardholder data. The Compliance Server acts as the central hub for monitoring and secure data relay, reducing the need for complex direct connections between other sensitive systems. The continued segmentation of the Internal Application Server, even while offline, ensures that its reintroduction into the network will be secure and adhere to the established security posture.

Hardening the Web Server and Database Server

Web Server Security Enhancements

The Web Server (192.168.10.10) plays a critical role in customer-facing operations. Initial assessment revealed vulnerabilities related to SSL/TLS settings, logging, and susceptibility to SQL injection and XSS.

Vulnerability Test Results:

- **SQL Injection Test (SQLi):**
    - Command: `curl -u admin:admin -d "name=Test OR 1=1 --&email=test@test.com&credit_card=1234&cvv=234" http://192.168.10.10/checkout.php`
    - Observation: The command likely returned output indicating successful manipulation of the database query (e.g., displaying all records or an unexpected result), confirming the SQL injection vulnerability.
    - **Impact:** This vulnerability allows attackers to bypass authentication, extract sensitive data (including credit card numbers if stored), modify data, or even gain control over the database.
- **Cross-Site Scripting (XSS) Test:**
    - Payload: `<script>alert('XSS')</script>` entered into a form field (e.g., a comment or name field).
    - Observation: The JavaScript `alert('XSS')` executed in the browser when viewing the page that displayed the submitted content, confirming the XSS vulnerability.
    - **Impact:** XSS allows attackers to inject malicious scripts into web pages viewed by other users. This can lead to session hijacking, defacement of websites, redirection to malicious sites, or stealing user credentials.

Security Enhancements Applied:

- **SSL/TLS Settings:**
    - **Improvement:** Enabled HTTPS with strong TLS protocols (TLS 1.2 or 1.3) and disabled weaker versions (SSLv2, SSLv3, TLS 1.0, TLS 1.1). Configured modern, strong cipher suites and disabled weak ciphers. Ensured proper certificate chain installation and renewal procedures.
    - **Implementation:** Edited the web server's configuration file (e.g., Apache's `httpd-ssl.conf` or Nginx's `nginx.conf`) to enforce these settings.

- **Logging:**
    - **Improvement:** Enabled comprehensive web server access and error logging. Configured logs to include source IP, timestamps, request details, and user-agent information. Ensured logs are securely stored, regularly reviewed, and forwarded to the Compliance Server (Wazuh agent) for centralized monitoring.
    - **Implementation:** Verified `CustomLog` and `ErrorLog` directives in web server configuration, ensured log rotation, and configured Wazuh agent to collect these logs.
- **Mitigating SQL Injection:**
    - **Improvement:** Implemented parameterized queries (prepared statements) for all database interactions. Input validation and sanitization were also applied to user-supplied data before it is used in SQL queries. Escaped special characters where prepared statements are not feasible (e.g., when building dynamic SQL, though this is generally discouraged).
    - **Implementation:** Modified the application code (e.g., `checkout.php` and other relevant scripts) to use PDO with prepared statements for MySQL interactions instead of direct string concatenation.
- **Mitigating XSS:**
    - **Improvement:** Implemented output encoding for all user-supplied data before it is rendered in the web application. This converts potentially malicious characters into their HTML entities, preventing them from being executed as scripts.
    - **Implementation:** Modified the application code to use appropriate encoding functions (e.g., `htmlspecialchars()` in PHP) for all user-generated content displayed on the web pages.

Did Nessus/Wazuh Scans Detect These Vulnerabilities?

- **Nessus:** Yes, Nessus, being a vulnerability scanner, would likely have detected common web application vulnerabilities like SQL injection and XSS if configured with appropriate web application scanning policies. It typically identifies these by analyzing application responses to various payloads.
- **Wazuh:** Wazuh, primarily an HIDS/SIEM, would detect anomalous behavior or specific attack patterns if rules are configured for them. It might detect attempts at SQL injection or XSS (e.g., specific malicious strings in logs) if the web server logs these attempts and Wazuh has rules to trigger alerts on them. However, it

wouldn't actively "scan" for these vulnerabilities in the same way Nessus does. It's more reactive.

How Identified and Mitigated Manually:

The vulnerabilities (SQLi and XSS) were identified manually by directly executing the provided test commands from the Desktop VM. This manual testing allowed for a direct observation of the vulnerability's exploitability. Mitigation was then applied by directly modifying the web server's configuration files (for SSL/TLS and logging) and the application's source code (for SQLi and XSS prevention) on the Web Server VM. This manual, direct interaction with the code and configurations is often necessary for specific application-level vulnerabilities that automated scanners might flag but not fully resolve.

Database Server Security Enhancements

The Database Server (192.168.20.10) stores sensitive PCI-related data, making its security paramount.

Security Enhancements Applied:

- **Access Control Policies:**
    - **Improvement:** Implemented the principle of least privilege for database users. Created specific users for each application (e.g., a separate user for the web application, compliance server backup process) with only the minimum necessary permissions (e.g., SELECT, INSERT, UPDATE, DELETE on specific tables, not DDL privileges). Removed or disabled default/unused accounts.
    - **PCI DSS Alignment:** Directly addresses PCI DSS Requirement 7.1 and 7.2 ("Restrict access to cardholder data by business need-to-know" and "Assign access rights based on individual’s job function and need to know").
- **Encryption Settings:**
    - **Improvement:** Enabled encryption for data in transit (SSL/TLS for MySQL connections) to protect sensitive data during communication between the application and the database. If sensitive data were stored at rest, disk-level or database-level encryption would be considered.
    - **PCI DSS Alignment:** Addresses PCI DSS Requirement 4.1 ("Use strong cryptography and security protocols to protect sensitive cardholder data

during transmission over open, public networks") and indirectly supports Requirement 3.4 ("Render PAN unreadable anywhere it is stored").

- **Logging Configurations:**
    - **Improvement:** Enabled comprehensive database auditing and logging. Configured MySQL to log all successful and failed access attempts, changes to database structure (DDL), and potentially sensitive queries (though performance impact must be considered). Configured logs to be immutable, rotated, and forwarded to the Compliance Server (Wazuh agent) for centralized security monitoring and analysis.
    - **PCI DSS Alignment:** Directly addresses PCI DSS Requirement 10 ("Log and monitor all access to network resources and cardholder data") and Requirement 10.3 ("Implement automated audit trails for all system components to reconstruct events").

Summary of Identified Vulnerabilities and Applied Security Enhancements:

| Server | Identified Vulnerabilities | Applied Security Enhancements during the day. This type of vulnerability can be very serious if not handled properly. This vulnerability can be exploited by an attacker to execute arbitrary SQL queries on the web server.

Nessus/Wazuh Scan Results

Here's a summary of potential findings from Nessus and Wazuh scans, assuming some common vulnerabilities on a Metasploitable 2 VM (which often serves as a good analogy for vulnerable systems):

Nessus Scan Results:

Nessus, a comprehensive vulnerability scanner, would likely report several critical findings on a system with known vulnerabilities, such as the Web Server and Database Server before hardening. Examples of critical findings from a Nessus scan could include:

- **Service Version Disclosure:** Many services (Apache, MySQL, SSH) might be running with easily identifiable version numbers, allowing attackers to quickly look up known exploits for those versions.
- **Default Credentials:** If any services (like FTP, SSH, or web applications) were still using default or weak credentials (e.g., `admin/admin`), Nessus would flag these immediately.
- **Open Ports and Unnecessary Services:** Nessus would enumerate all open ports and identify running services. If services like Telnet, unencrypted FTP, or

older Samba versions were found, they would be flagged as high risk due to their inherent insecurity or vulnerabilities.
- **Outdated Software:** Nessus would identify if Apache, MySQL, PHP, or the operating system itself had unpatched vulnerabilities, providing CVE references.
- **Web Application Vulnerabilities (Pre-Hardening):**
    - **SQL Injection:** Nessus's web application scanning capabilities would detect the SQL injection vulnerability on the web server by sending various payloads and analyzing responses.
    - **Cross-Site Scripting (XSS):** Similar to SQLi, Nessus would identify XSS vulnerabilities by injecting scripts and observing their execution in simulated browser environments.
- **Missing Security Headers:** Lack of HTTP security headers (like HSTS, X-Content-Type-Options, X-Frame-Options) would be reported as medium to low severity but important for overall security.
- **Weak SSL/TLS Configuration:** Prior to hardening, Nessus would flag the use of weak TLS protocols (TLS 1.0, 1.1) or weak cipher suites.

Wazuh Alerts (Real-time Monitoring):

Wazuh, functioning as an HIDS/SIEM, would provide real-time alerts based on its agents deployed on the servers and its rule-set. During the course of the assessment and assuming some initial vulnerabilities, Wazuh might generate alerts for:

- **Failed Authentication Attempts:** Numerous failed login attempts to SSH, web applications, or MySQL from suspicious IPs (brute-force attacks).
- **Known Attack Patterns:** If the web application was being actively probed, Wazuh rules might trigger on suspicious URLs, SQLi payloads in web server logs, or XSS attempts.
- **File Integrity Monitoring (FIM) Alerts:** Unauthorized changes to critical system files, web application files, or database configuration files.
- **Rootkit Detection:** Alerts if any suspicious processes or rootkits are detected on the system.
- **System Audit Events:** Alerts for changes to user accounts, privilege escalation attempts, or modifications to firewall rules (if Wazuh is configured to monitor `/etc/sysconfig/iptables` or similar).
- **Compliance Policy Violations:** Wazuh can be configured with PCI DSS policies. It would alert on deviations from these policies, e.g., if a system tries to communicate with an unauthorized network, or if an insecure service is started.

Impact on Network Security:

Critical findings from Nessus and real-time alerts from Wazuh have significant implications for network security:

- **Data Breach Risk:** SQL injection and weak database access controls directly expose sensitive cardholder data, leading to a high risk of data breaches and non-compliance with PCI DSS Requirement 3 (Protect stored cardholder data).
- **System Compromise:** Weak credentials, outdated software, and open ports provide attackers with multiple entry points to compromise systems, leading to unauthorized access, data theft, or malware deployment.
- **Loss of Confidentiality, Integrity, and Availability:** Exploitable vulnerabilities can lead to loss of confidentiality (data theft), integrity (data manipulation, website defacement), and availability (DoS attacks).
- **Compliance Violations:** The presence of these vulnerabilities directly violates numerous PCI DSS requirements, leading to fines, reputational damage, and loss of business.
- **Insider Threat Risk:** Unmonitored systems and weak logging make it difficult to detect and respond to insider threats, as demonstrated by the incident with the Internal Application Server.

Automated vs. Manual Compliance Audits

Advantages and Disadvantages of Automated Compliance Tools Compared to Manual Audits:

Feature	Automated Compliance Tools (e.g., Nessus, Wazuh)	Manual Audits (Checklists, Interviews, Manual Reviews)

Advantages	- **Speed and Efficiency:** Can scan large networks quickly, identify vulnerabilities in minutes or hours. - **Consistency:** Provides standardized and repeatable assessments, reducing human error. - **Scalability:** Easily deployed across numerous systems. - **Comprehensive Coverage:** Can check for thousands of known vulnerabilities and misconfigurations. - **Real-time Monitoring:** (Wazuh) Continuous monitoring and alerting for ongoing threats. - **Cost-Effective:** Lower long-term costs compared to continuous manual efforts for basic checks.	- **Depth and Context:** Can identify logical flaws, business process vulnerabilities, and complex issues that automated tools miss. - **Qualitative Assessment:** Allows for nuanced interpretation of policies, interviews with personnel, and assessment of human factors. - **Customization:** Can be tailored to specific organizational contexts, unique applications, or complex architectures. - **Validation:** Can verify the effectiveness of controls that automated tools only identify the presence of (e.g., security awareness training). - **Zero-Day Detection:** Can sometimes uncover novel vulnerabilities or exploit chains through creative manual testing.

Disadvantages	- **False Positives/Negatives:** Can generate false alarms or miss complex, subtle vulnerabilities. - **Limited Context:** Lacks understanding of business logic, human behavior, or specific application workflows. - **Snapshot in Time:** (Scanners) Provide a view of vulnerabilities at a specific point; not continuous without re-scanning. - **Requires Interpretation:** Raw scan results often need skilled human analysis to prioritize and remediate. - **Tool Dependency:** Effectiveness is limited by the tool's database and capabilities. - **Resource Intensive (Initial Setup):** Can require significant configuration to avoid disrupting operations.	- **Time-Consuming:** Manual audits are inherently slower, especially for large infrastructures. - **Human Error/Inconsistency:** Results can vary based on auditor skill, experience, and attention to detail. - **High Cost:** Requires significant human resources, making it expensive for frequent or large-scale audits. - **Limited Scale:** Difficult to apply consistently across thousands of systems. - **Lack of Real-time:** Provides a periodic snapshot, not continuous monitoring. - **Skill Dependency:** Requires highly skilled and experienced auditors.

Export to Sheets

When is it more effective to rely on automated scanning versus manual verification?

- **Automated Scanning is More Effective When:**
    - **Initial Discovery:** For broad network and host-level vulnerability assessments to quickly identify common misconfigurations, missing patches, and known vulnerabilities across a large asset base.
    - **Regular, Repetitive Checks:** For continuous monitoring of compliance settings (e.g., firewall policy adherence, password complexity) or detecting the introduction of new, known vulnerabilities.

    - **Large-Scale Environments:** When auditing hundreds or thousands of systems where manual checks would be impractical.
    - **Baseline Assessments:** To establish a security baseline against which future changes can be compared.
    - **Specific Technical Checks:** For validating technical configurations like SSL/TLS settings, open ports, and software versions.
- **Manual Verification is More Effective When:**

    - **Business Logic Flaws:** Assessing vulnerabilities specific to custom applications where the flaw lies in the application's unique business logic (e.g., authorization bypasses based on workflow).
    - **Complex Architectural Reviews:** Evaluating the overall security design of a complex system, including data flow, trust boundaries, and compensating controls.
    - **Policy and Procedure Review:** Verifying that security policies are appropriate, documented, communicated, and actually followed by personnel.
    - **Social Engineering/Human Factors:** Testing human susceptibility to phishing, pretexting, or other social engineering attacks.
    - **Penetration Testing:** Simulating real-world attacks to exploit vulnerabilities and chain them together to assess the overall security posture and identify previously unknown weaknesses.
    - **Forensic Analysis/Incident Response:** Deep-dive investigation into specific incidents.
    - **Validating Automated Findings:** Manually confirming the existence and severity of critical findings reported by automated tools to reduce false positives.

How would a hybrid approach improve vulnerability detection and PCI DSS compliance?

A hybrid approach, combining both automated and manual methods, represents the most effective strategy for robust vulnerability detection and comprehensive PCI DSS compliance:

1. **Automated for Breadth and Speed, Manual for Depth and Context:**

    - Automated tools (Nessus, Wazuh) can quickly and efficiently cover the vast majority of common technical vulnerabilities across the entire IT estate.

    - Manual audits and penetration tests then provide deeper insights into specific critical systems, complex application logic, and human factors that automated tools cannot assess.
2. **Continuous Monitoring with Periodic Deep Dives:**
    - Wazuh provides continuous, real-time security monitoring and compliance posture management, alerting to deviations as they occur.
    - Nessus provides scheduled, comprehensive vulnerability scans.
    - These are complemented by periodic manual penetration tests (e.g., annually or after significant changes) and internal/external audits to provide an exhaustive review and validate the effectiveness of ongoing controls.
3. **Validation and Prioritization:**
    - Automated tools flag many potential vulnerabilities. Manual review helps to validate these findings, eliminate false positives, and prioritize remediation efforts based on actual risk and business impact, integrating with PCI DSS requirements.
4. **Policy and Process Verification:**
    - Automated tools can verify technical configurations against PCI DSS requirements. Manual audits are essential for verifying the implementation and effectiveness of procedural controls, such as incident response plans, security awareness training, and change management processes, which are critical for PCI DSS.
5. **Adapting to New Threats:**
    - While automated tools are updated with new vulnerability definitions, manual efforts, including threat intelligence analysis and ethical hacking, are crucial for identifying zero-day vulnerabilities and adapting to evolving attack techniques.

By leveraging the strengths of both automated efficiency and manual analytical depth, an organization can achieve a more complete and resilient security posture, ensuring continuous adherence to PCI DSS compliance standards.

Reflection and Future Improvements

Effectiveness of Network Security Implementation

The implemented segmentation rules have effectively enforced the isolation of sensitive systems as per the design requirements.

- **Proper Isolation:** The Web Server, Database Server, and Internal Application Server are logically isolated from each other. Direct communication paths that could lead to lateral movement or unauthorized access to sensitive data (e.g., Web Server to Database Server) have been successfully blocked.
- **Firewall Impact on Security and Access Control:** The VyOS firewall rules have significantly enhanced security by implementing a default-deny posture. Only explicitly authorized traffic is permitted, which is a fundamental security best practice. This granular control has directly improved access control, ensuring that only necessary communication occurs, particularly for the CDE. The rules align directly with PCI DSS requirements for network segmentation and traffic filtering.

Challenges and Adjustments

Challenges Encountered:

- **Initial Connectivity Issues:** When first implementing the `drop all` rules or specific `drop` rules, there were potential temporary connectivity disruptions for essential services if the `accept` rules were not perfectly sequenced or comprehensive enough. For example, ensuring the Compliance Server could reach all systems for monitoring after initial broad `drop` rules.
- **Maintaining Internal Server Rules:** Keeping the Internal Application Server's segmentation rules intact while it was offline required careful planning to ensure no unintended connectivity was introduced or removed, especially when implementing broader `drop` rules that might override existing specific `accept` rules for this server.
- **Rule Ordering Complexity:** In firewall configurations, the order of rules is critical. A broad `accept` rule placed before a specific `drop` rule can negate the intended denial. This required careful consideration and testing.

Resolution of Connectivity or Security Issues:

- **Incremental Implementation:** Firewall rules were implemented and tested incrementally. Instead of applying all rules at once, a phased approach was used, adding rules layer by layer (e.g., first basic `accept established`, then specific `drop` rules, then specific `accept` rules) and verifying connectivity at each step.
- **Logging and Monitoring:** During the implementation phase, increased logging on the VyOS router was crucial to identify dropped packets and troubleshoot

connectivity issues. Wazuh monitoring on the Compliance Server helped in real-time detection of blocked traffic that should have been allowed or vice-versa.
- **Review and Refinement:** Post-implementation, a thorough review of the rule set was conducted to ensure logical correctness and adherence to the principle of least privilege. Minor adjustments to source/destination IPs or port ranges were made as needed to fine-tune legitimate traffic flows while maintaining isolation.

Future Security Enhancements

- **Intrusion Prevention System (IPS):** Integrating an IPS (e.g., using Snort or Suricata on VyOS if supported, or a dedicated appliance) would provide an additional layer of defense. It could actively block known attack patterns (e.g., SQL injection attempts, XSS exploitation, brute-force attacks) in real-time, even before they reach the application. This moves from passive detection to active prevention.
- **Web Application Firewall (WAF):** Deploying a WAF in front of the Web Server would significantly enhance its protection against application-layer attacks (SQL injection, XSS, CSRF, etc.). A WAF can inspect HTTP traffic, filter malicious requests, and provide virtual patching for known vulnerabilities without modifying application code. This is a critical control for PCI DSS Requirement 6.6.
- **Stronger Authentication and Multi-Factor Authentication (MFA):** Implement MFA for all administrative access to the Web Server, Database Server, and VyOS router. For the database, consider strong password policies, regular rotation, and potentially certificate-based authentication for critical services. This aligns with PCI DSS Requirement 8.3 and 8.5.
- **Data Encryption at Rest:** Implement encryption for sensitive data stored on the Database Server's disk. This could be achieved through file-level, disk-level, or database-level encryption (e.g., MySQL Transparent Data Encryption). This is a strong control for PCI DSS Requirement 3.4.
- **Principle of Least Functionality:** Review all servers to disable unnecessary services and applications. This reduces the attack surface and minimizes potential entry points for attackers.
- **Automated Configuration Management:** Implement tools like Ansible or SaltStack to manage server configurations. This ensures consistency, reduces human error, and allows for rapid deployment of secure baselines and updates.
- **Security Awareness Training:** Continuous security awareness training for all employees, especially those with access to sensitive systems or data, is crucial to mitigate insider threats and human-related vulnerabilities.

Logging and Monitoring Integration for Better Security Analysis

- **Centralized Log Management and SIEM Enhancement:**
    - **Current:** Wazuh is acting as a SIEM.
    - **Improvement:** Ensure all relevant logs (Web Server access/error logs, Database Server audit logs, VyOS firewall logs, SSH logs, OS security logs) are consistently forwarded to Wazuh.
    - **Enrichment:** Configure Wazuh to enrich logs with threat intelligence feeds (e.g., known malicious IPs).
    - **Correlation Rules:** Develop more sophisticated correlation rules within Wazuh to identify complex attack patterns that span multiple systems or log sources (e.g., a failed login on the web server followed by an SSH brute-force attempt from the same IP).
    - **Custom Alerts:** Create custom alerts for specific high-risk events (e.g., excessive failed SQL queries, large data transfers, changes to critical security files).
    - **Dashboarding and Reporting:** Utilize Wazuh's dashboarding capabilities to visualize security events, compliance posture, and trends, providing actionable insights for security analysts and management.
- **Automated Alerting and Incident Response Integration:**
    - Integrate Wazuh alerts with an incident response platform or ticketing system. This ensures that critical security incidents are immediately triaged, assigned, and responded to in a timely manner, adhering to PCI DSS Requirement 12.10.
- **Regular Log Review and Forensics Capabilities:**
    - Establish a formal process for regular review of logs.
    - Ensure logs are stored securely and for the required retention period (PCI DSS Requirement 10.7 and 10.9).
    - Maintain the ability to perform forensic analysis on logs in the event of a security incident, using them to reconstruct events and identify the root cause.

By implementing these future enhancements and optimizing logging and monitoring, the organization can achieve a more mature, proactive, and resilient security posture, significantly improving its ability to detect, prevent, and respond to cyber threats while maintaining robust PCI DSS compliance.