

VULNERD PROJECT

Detection Playbook

Purpose

The purpose of the detection playbook is basically a guide for what we want to catch and how we react to it. Instead of treating every Nessus finding or log entry as random noise, the playbook lists the key patterns we care about (for example, “web app leaking version info” or “failed SSH credentials”), explains why they matter, and describes what should happen when we see them. It turns raw technical alerts into clear “if this happens, then do that” rules. This helps the team stay consistent, focus on the most important signals, and respond faster when something suspicious shows up in our scans.

Structure

Each detection entry in this playbook follows a consistent structure:

- Detection name
- Related Nessus plugins (by name or ID)
- What it means (threat / risk)
- Trigger logic (in plain IF/THEN terms)
- Priority (Low / Medium / High / Critical)
- Recommended response

Data Sources

This playbook was built from findings across multiple scan types and manual attack testing performed against the VULNERD lab environment (DVWA on Raspberry Pi 5, accessed via Tailscale VPN on port 8081). The data sources include:

- **Basic Network Scan (Credentialed via SSH):** 1 Critical, 2 High, 6 Medium, 1 Low, 72 Info findings
- **Web Application Tests Scan:** 0 Critical, 0 High, 1 Medium, 1 Low, 24 Info findings
- **Advanced Dynamic Scan:** 1 Critical, 0 High, 0 Medium, 0 Low, 11 Info findings
- **Manual Attack Testing:** Brute Force, Command Injection, SQL Injection, XSS (Reflected, Stored, DOM), File Inclusion, File Upload — all successfully exploited on DVWA (Security Level: Low)

Detections

Detection 1 — Critical System Library Vulnerability (libnss3)

Field	Details
Plugins	300454 — Debian dsa-6149 : libnss3 - security update
What it means	The Network Security Services (NSS) library on the host has a known critical vulnerability. NSS handles SSL/TLS and cryptographic operations. A compromised NSS library could allow an attacker to intercept encrypted communications, forge certificates, or execute arbitrary code. This was detected by both the Basic Network Scan and the Advanced Dynamic Scan, confirming it is a real and persistent issue.
Trigger logic	IF a credentialed scan detects that the installed libnss3 package version is older than the version specified in Debian Security Advisory DSA-6149, THEN raise a Critical alert.
Priority	Critical
Recommended response	Apply the Debian security update immediately using: <code>sudo apt update && sudo apt upgrade libnss3</code> . Verify the update was applied by re-scanning. This library is fundamental to encrypted communications on the system.

Detection 2 — Outdated System Libraries with Known Vulnerabilities

Field	Details
Plugins	299347 — Debian dsa-6138 : libpng-dev - security update (High) 299381 — Debian dsa-6140 : gnutls-bin - security update (Medium)
What it means	Multiple system libraries (libpng for image processing, GnuTLS for secure communications) have known vulnerabilities with available patches. These outdated libraries could be exploited to cause denial of service, information disclosure, or in some cases remote code execution. In a small organization without dedicated security staff, unpatched libraries are one of the most common attack vectors.
Trigger logic	IF a credentialed scan identifies installed packages with versions older than those specified in Debian Security Advisories (DSA-6138, DSA-6140), THEN raise an alert at the highest severity level among the affected packages.
Priority	High
Recommended response	Run a full system update: <code>sudo apt update && sudo apt upgrade -y</code> . Establish a regular patching schedule (at minimum monthly). Consider enabling unattended-upgrades for automatic security patches on the Raspberry Pi.

Detection 3 — Vim Editor Multiple Vulnerabilities

Field	Details
Plugins	301247 — Vim < 9.2.0073 Command Injection (High) 298656 — Vim < 9.1.2132 Buffer Overflow (Medium) 301246 — Vim < 9.2.0074 Heap-based Buffer Overflow (Medium) 301248 — Vim < 9.2.0075 Heap-based Buffer Underflow (Medium) 301509 — Vim < 9.2.0076 Heap-based Buffer Overflow and OOB Read (Medium) 301510 — Vim < 9.2.0077 Heap-based Buffer Overflow (Medium) 301508 — Vim < 9.2.0078 Stack-Buffer-Overflow (Low)
What it means	The installed version of Vim has seven known vulnerabilities including command injection, multiple buffer overflows, and a buffer underflow. While Vim is a text editor typically used locally, these vulnerabilities could be exploited if an attacker tricks a user into opening a maliciously crafted file. The command injection vulnerability (High severity) is particularly concerning as it could allow arbitrary command execution on the host.
Trigger logic	IF a credentialed scan detects Vim version older than 9.2.0078 AND reports any of the listed plugin IDs, THEN flag the host for immediate patching with priority based on the highest severity finding (High).
Priority	High
Recommended response	Update Vim to the latest available version: <code>sudo apt update && sudo apt install vim</code> . If Vim is not actively needed on the server, consider removing it entirely to reduce the attack surface. Never open untrusted files in Vim on production systems.

Detection 4 — Web Application Exposed on Non-Standard Port

Field	Details
Plugins	10107 — HTTP Server Type and Version 48204 — Apache HTTP Server Version 48243 — PHP Version Detection 22964 — Service Detection
What it means	A web application (DVWA) is exposed on port 8081 instead of the standard 80/443. The server leaks version information through HTTP headers, including the Apache and PHP versions. This gives attackers a precise fingerprint of the software stack, allowing them to search for version-specific exploits. The non-standard port does not provide security through obscurity, as port scanners easily discover all listening services.
Trigger logic	IF a host has an HTTP service on any non-standard port (not 80/443) AND Nessus reports HTTP Server Type and Version or PHP Version Detection for that host, THEN raise an alert.
Priority	Medium
Recommended response	Restrict access to port 8081 to VPN/Tailscale only (already in place). Review Apache and PHP configurations to suppress version banners

	(ServerTokens Prod, ServerSignature Off, expose_php = Off). Add security headers to HTTP responses.
--	---

Detection 5 — OS and Network Fingerprinting Exposure

Field	Details
Plugins	209654 — OS Fingerprints Detected 11936 — OS Identification 25220 — TCP/IP Timestamps Supported 10287 — Traceroute Information
What it means	An external scanner can identify the operating system, kernel version, and network path to the host. TCP timestamps and traceroute results make it easier for attackers to estimate uptime, map the network topology, and select tailored exploits for the specific OS version. The scan identified the system as Linux Kernel 2.6, which may indicate fingerprinting inaccuracy but still reveals the OS family.
Trigger logic	IF a host has both OS Fingerprints Detected and TCP/IP Timestamps Supported in a single scan, THEN flag it as high reconnaissance visibility.
Priority	Medium
Recommended response	Consider blocking or limiting traceroute and ICMP-based fingerprinting from untrusted networks. Disable TCP timestamps if not needed (net.ipv4.tcp_timestamps=0 in sysctl.conf). Treat this host as requiring extra monitoring since it is easy to profile.

Detection 6 — Web Application Clickjacking Vulnerability

Field	Details
Plugins	85582 — Web Application Potentially Vulnerable to Clickjacking 50344 — Missing or Permissive Content-Security-Policy frame-ancestors HTTP Response Header 50345 — Missing or Permissive X-Frame-Options HTTP Response Header
What it means	The web application does not set X-Frame-Options or Content-Security-Policy frame-ancestors headers. This means the application pages could be embedded in a hidden iframe on an attacker-controlled site, tricking users into clicking on elements they cannot see. This is known as a clickjacking or UI redress attack and could lead to unauthorized actions being performed under the user's authenticated session.
Trigger logic	IF a web application scan detects that HTTP responses lack both X-Frame-Options and Content-Security-Policy frame-ancestors headers AND the page contains clickable elements (forms, buttons), THEN raise an alert.
Priority	Medium

Recommended response	Configure the web server to send X-Frame-Options: DENY or SAMEORIGIN header. Additionally, implement Content-Security-Policy with frame-ancestors directive. In Apache, add: Header always set X-Frame-Options DENY and Header always set Content-Security-Policy "frame-ancestors 'none'" to the configuration.
-----------------------------	--

Detection 7 — Cleartext Credential Transmission

Field	Details
Plugins	26194 — Web Server Transmits Cleartext Credentials
What it means	The DVWA login page transmits username and password over unencrypted HTTP (port 8081) rather than HTTPS. An attacker on the same network segment or performing a man-in-the-middle attack could intercept login credentials in plaintext. Even though access is restricted to the Tailscale VPN, defense in depth requires encryption at the application layer as well.
Trigger logic	IF a web application scan detects HTML forms with password input fields that submit to an HTTP (not HTTPS) URL, THEN raise an alert.
Priority	Low
Recommended response	Configure TLS/SSL on the web server so DVWA is served over HTTPS. Obtain a certificate (self-signed is acceptable for lab environments) and configure Apache to redirect all HTTP traffic to HTTPS. This protects credentials even if the VPN tunnel were compromised.

Detection 8 — SSH Weak HMAC Algorithms Enabled

Field	Details
Plugins	153588 — SSH SHA-1 HMAC Algorithms Enabled
What it means	The SSH server accepts connections using SHA-1 based HMAC (Hash-based Message Authentication Code) algorithms. SHA-1 is considered cryptographically weak and vulnerable to collision attacks. While exploiting this in a real SSH session is currently difficult, it represents a deviation from security best practices and could become more exploitable as computing power increases.
Trigger logic	IF an SSH server supports HMAC algorithms based on SHA-1 (such as hmac-sha1 or hmac-sha1-etm), THEN flag it as a configuration weakness.
Priority	Low
Recommended response	Edit /etc/ssh/sshd_config to restrict HMAC algorithms to SHA-256 or SHA-512 variants only. Add: MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm@openssh.com,hmac-sha2-

	256,hmac-sha2-512. Restart the SSH service and verify with a new scan.
--	--

Detection 9 — robots.txt Information Disclosure

Field	Details
Plugins	10302 — Web Server robots.txt Information Disclosure
What it means	The robots.txt file on port 8081 is publicly accessible and could reveal hidden or sensitive directory paths. While robots.txt is intended to guide search engine crawlers, attackers routinely check it to discover administrative interfaces, backup directories, or internal application paths that the site owner intended to keep hidden.
Trigger logic	IF robots.txt Information Disclosure is present AND the robots.txt file references any directory under /admin, /backup, /private, or other sensitive paths, THEN raise an alert for sensitive paths exposed.
Priority	Low (higher if sensitive paths are listed)
Recommended response	Review robots.txt and remove references to sensitive paths. Use proper access controls (authentication, IP restrictions) instead of relying on robots.txt to hide resources. In a lab environment, consider removing robots.txt entirely.

Manual Testing Detections

The following detections are based on vulnerabilities that were **successfully exploited through manual testing** but were **not detected by any of the Nessus scans**. This gap highlights the limitations of automated scanning tools and reinforces why manual testing and human expertise remain essential components of a comprehensive security program. These findings directly support Vulnerd's purpose of combining automated detection with human judgment.

Detection 10 — Brute Force Authentication Attack

Field	Details
Plugins	Not detected by Nessus — Identified through manual testing
What it means	The DVWA login form has no rate limiting, account lockout, or CAPTCHA protection. An attacker can submit unlimited password guesses without any defensive mechanism intervening. During manual testing, a curl-based brute force loop successfully tried multiple passwords and gained access using the credential admin/password. This represents a fundamental authentication weakness.
Trigger logic	IF web server logs show more than 5 failed login attempts from the same source IP within a 60-second window, followed by a successful login, THEN raise a brute force alert.
Priority	High
Recommended response	Implement account lockout after 5 failed attempts (15-minute lockout). Add CAPTCHA after 3 failed attempts. Implement rate limiting on the login endpoint. Monitor authentication logs for patterns of failed-then-successful logins. Enforce strong password policies.

Detection 11 — OS Command Injection via Web Application

Field	Details
Plugins	Not detected by Nessus — Identified through manual testing
What it means	The DVWA Command Injection module passes user input directly to a system shell command without sanitization. During testing, appending system commands with a semicolon (;) after a valid IP address resulted in full command execution on the server. The whoami command revealed the web server runs as www-data, cat /etc/passwd exposed all system accounts, and uname -a revealed the full kernel version (Linux 6.12.47 ARM64). This is a critical vulnerability that gives an attacker remote code execution on the server.
Trigger logic	IF web application logs or a WAF detects input containing shell metacharacters (;, , &&, , \$(), backticks) in parameters expected to contain simple values (IP addresses, filenames), THEN raise a critical alert.
Priority	Critical

Recommended response	Implement strict input validation using allowlists (e.g., only accept valid IP address formats). Never pass user input directly to system commands. Use parameterized APIs instead of shell execution. Deploy a Web Application Firewall (WAF) to detect and block command injection patterns.
-----------------------------	--

Detection 12 — SQL Injection with Data Exfiltration

Field	Details
Plugins	Not detected by Nessus — Identified through manual testing
What it means	The DVWA SQL Injection module passes user input directly into SQL queries without parameterization or sanitization. During testing, the payload 1' OR 1=1# successfully dumped the entire user table (5 accounts: admin, Gordon Brown, Hack Me, Pablo Picasso, Bob Smith). More critically, the UNION-based payload 1' UNION SELECT user, password FROM users# extracted all usernames and their MD5 password hashes directly from the database. This represents a catastrophic data breach scenario where an attacker gains complete access to credential data.
Trigger logic	IF web application logs or a WAF detects SQL keywords (UNION, SELECT, OR, AND, DROP, INSERT) combined with SQL syntax characters (single quotes, semicolons, comment markers like # or --) in user input fields, THEN raise a critical alert.
Priority	Critical
Recommended response	Use parameterized queries (prepared statements) for all database interactions. Implement input validation to reject SQL metacharacters. Deploy a WAF with SQL injection detection rules. Apply principle of least privilege to database accounts — the web application should not have access to query user credentials directly. Store passwords using strong hashing (bcrypt, argon2) instead of MD5.

Detection 13 — Cross-Site Scripting (XSS) — Reflected, Stored, and DOM

Field	Details
Plugins	Not detected by Nessus — Identified through manual testing
What it means	Three types of XSS vulnerabilities were confirmed in DVWA. Reflected XSS: injecting <script>alert('XSS')</script> in the name field caused immediate script execution. Stored XSS: the same payload in the guestbook Message field was saved to the database and executes every time any user visits the page. DOM-based XSS: modifying the URL default parameter injected a script that executed entirely client-side. Stored XSS is the most dangerous as it persists and affects all visitors. These vulnerabilities could be used to steal session cookies,

	redirect users to malicious sites, or perform actions on behalf of authenticated users.
Trigger logic	IF web application logs or a WAF detects HTML tags (<script>, , <iframe>), JavaScript event handlers (onerror, onload, onclick), or encoded variants (%3Cscript%3E) in user-submitted input, THEN raise a high alert.
Priority	High
Recommended response	Implement output encoding (HTML entity encoding) for all user-supplied data rendered in web pages. Use Content-Security-Policy headers to prevent inline script execution. Implement input validation to reject HTML tags in text fields. For Stored XSS specifically, sanitize all data before writing to the database. Use a framework with built-in XSS protection.

Detection 14 — Local File Inclusion via Path Traversal

Field	Details
Plugins	Not detected by Nessus — Identified through manual testing
What it means	The DVWA File Inclusion module accepts a filename parameter in the URL without validating the path. By using path traversal sequences (../../../../etc/passwd), the server was tricked into reading and displaying the system password file. Additionally, the application leaked internal file paths through PHP warning messages (/var/www/html/dvwa/includes/dvwaPage.inc.php). This vulnerability allows an attacker to read any file on the server that the web server process has permission to access.
Trigger logic	IF web application logs or a WAF detects path traversal sequences (../, ..\, %2e%2e%2f) or references to system files (/etc/passwd, /etc/shadow, /proc/) in URL parameters or request bodies, THEN raise a high alert.
Priority	High
Recommended response	Validate and sanitize all file path parameters using an allowlist of permitted files. Use a base directory restriction to prevent traversal outside the web root. Disable PHP error display in production (display_errors = Off). Configure open_basedir in PHP to restrict file access to the application directory.

Detection 15 — Unrestricted File Upload

Field	Details
Plugins	Not detected by Nessus — Identified through manual testing
What it means	The DVWA File Upload module accepts file uploads with zero validation on file type, content, or extension. During testing, a PHP file

	containing executable code was uploaded and the server both accepted and executed it. This is a critical vulnerability because an attacker could upload a web shell (a PHP script that provides full command-line access to the server), effectively gaining persistent remote access that survives server restarts.
Trigger logic	IF web server logs show a file upload followed by an HTTP request to access the uploaded file in the uploads directory AND the uploaded file has an executable extension (.php, .asp, .jsp, .py, .sh), THEN raise a critical alert.
Priority	Critical
Recommended response	Implement strict file type validation using both extension checking and MIME type/magic byte verification. Store uploaded files outside the web root so they cannot be directly accessed via URL. Rename uploaded files to prevent execution. Set the uploads directory permissions to prevent script execution. Implement file size limits and scan uploads with antivirus.

Detection Summary

The following table provides a quick reference for all detections in this playbook, organized by priority.

#	Detection Name	Priority	Source	Key Plugin(s)
1	Critical System Library Vulnerability (libnss3)	Critical	Nessus	300454
11	OS Command Injection via Web Application		Manual	N/A
12	SQL Injection with Data Exfiltration		Manual	N/A
15	Unrestricted File Upload		Manual	N/A
2	Outdated System Libraries	High	Nessus	299347, 299381
3	Vim Editor Multiple Vulnerabilities		Nessus	301247 +6 more
10	Brute Force Authentication Attack		Manual	N/A
13	Cross-Site Scripting (XSS)		Manual	N/A
14	Local File Inclusion via Path Traversal		Manual	N/A
4	Web App Exposed on Non-Standard Port	Medium	Nessus	10107, 48204, 48243
5	OS and Network Fingerprinting Exposure		Nessus	209654, 25220
6	Clickjacking Vulnerability		Nessus	85582
7	Cleartext Credential Transmission	Low	Nessus	26194
8	SSH Weak HMAC Algorithms		Nessus	153588
9	robots.txt Information Disclosure		Nessus	10302

Key Insight: The Detection Gap

A significant finding from this exercise is that **6 out of 15 detections (40%) came exclusively from manual testing and were completely missed by all automated Nessus scans**. These 6 undetected vulnerabilities include 3 of the 4 Critical-priority findings (Command Injection, SQL Injection, and Unrestricted File Upload) and represent the most dangerous attack vectors in the environment.

This gap exists because Nessus excels at network-level and infrastructure vulnerability detection (outdated packages, misconfigurations, weak protocols) but has limitations when testing application-layer vulnerabilities that require authenticated crawling and active injection testing. Dedicated web application security testing tools (such as Burp Suite or OWASP ZAP) and manual penetration testing are essential to achieve comprehensive coverage.

This finding directly supports Vulnerd's core purpose: automated tools provide valuable continuous monitoring, but human expertise and judgment remain essential for identifying and responding to the most critical threats. A defense-in-depth approach that combines automated scanning, threat intelligence, manual testing, and AI-assisted reporting delivers the most complete security coverage for under-resourced organizations.