

LINUX SECURITY

part_4.sh



Author: aimery de La Roncière @ [in](#) aimerydlr
University: ECE Paris – Ing5 CYB
YEAR: 2025-2026



Table of contents

I.	Vulnerability assessment and Patch Management.....	4
I.1.	Technical Standards and Vulnerability Databases	4
I.2.	Automated vulnerability assessment – OpenVAS.....	20
I.3.	Automated vulnerability assessment – Nuclei.....	26
I.4.	Linux Package Manager	31
I.5.	Centralized Patch & Endpoint Management	38
II.	Further Resources and Recommended Readings	43



I. Vulnerability assessment and Patch Management

Vulnerability assessment and patch management are essential pillars of a proactive cybersecurity strategy. They ensure that known security weaknesses are identified, assessed, and remediate in a timely manner to reduce the risk of exploitation. This chapter presents an overview of the fundamental standards and databases that catalog vulnerabilities, explores leading automated vulnerability assessment tools, and covers best practices for managing software updates and IT assets.

I.1. Technical Standards and Vulnerability Databases

Common Weakness Enumeration (CWE)

- Created: 2006
- Maintained by: MITRE Corporation
- URL: <https://cwe.mitre.org/>

CWE is a formal catalog of common software weaknesses and programming errors. Unlike specific vulnerabilities, CWE focuses on the types of flaws that cause vulnerabilities, such as buffer overflows, cross-site scripting, or improper input validation. By identifying these generic weakness classes, developers and security analysts can understand root causes and improve secure coding practices. CWE is widely used to guide static analysis tools, security training, and vulnerability research.

Example use: When a vulnerability is reported in CVE, it often references a CWE to describe the underlying weakness category.

CWE-ID	CWE Name	Source
CWE-94	Improper Control of Generation of Code ('Code Injection')	NIST
CWE-1336	Improper Neutralization of Special Elements Used in a Template Engine	MITRE

Figure 1 - Related CWE for CVE-2025-47916

Source: <https://nvd.nist.gov/vuln/detail/CVE-2025-47916>



cwe.mitre.org/data/definitions/94.html

CWE Common Weakness Enumeration

A community-developed list of SW & HW weaknesses that can become vulnerabilities

Home > CWE List > CWE-94: Improper Control of Generation of Code ('Code Injection') (4.1.2)

Weakness ID: 94
Vulnerability Mapping: ALLOWED (with careful review of mapping notes)
Abstraction: Base

View customized information: Conceptual Operational Mapping Friendly Complete Custom

Description

The product constructs all or part of a code segment using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the syntax or behavior of the intended code segment.

Alternate Terms
Code Injection

Common Consequences

Impact	Details
Bypass Protection Mechanism	Scope: Access Control In some cases, injectable code controls authentication; this may lead to a remote vulnerability.
Gain Privileges or Assume Identity	Scope: Access Control Injected code can access resources that the attacker is directly prevented from accessing.
Execute Unauthorized Code	Scope: Integrity, Confidentiality, Availability When a product allows a user's input to contain code syntax, it might be possible for an attacker to craft the code in such a way that it will alter the intended control flow of the product. As a result,

Figure 2 - CWE-94: Improper Control of Generation of Code ('Code Injection')

Source: <https://cwe.mitre.org/data/definitions/94.html>



Common Vulnerabilities and Exposures (CVE)

- Created: 1999
- Maintained by: MITRE Corporation
- URL: <https://www.cve.org/>

CVE is the industry-standard catalog that assigns unique identifiers to individual, publicly disclosed cybersecurity vulnerabilities and exposures. Each CVE entry (e.g., CVE-2025-47916) represents a specific vulnerability in a software product or hardware device. CVE serves as a global naming standard, enabling security tools and vendors to share consistent information about vulnerabilities. CVE itself does not provide detailed vulnerability metrics but acts as an index for referencing vulnerability reports, advisories, and patches.

The screenshot shows the CVE-2025-47916 record page. At the top, the header includes the CVE 25th anniversary logo, navigation links for About, Partner Information, Program Organization, Downloads, Resources & Support, and a Report/Request button. The main content area displays the CVE ID (CVE-2025-47916), status (PUBLISHED), and links to View JSON and User Guide. A "Required CVE Record Information" section contains details about the CNA (MITRE Corporation), publication date (2025-05-16), and update date (2025-05-16). It also lists the CWE (CWE-1336: Improper Neutralization of Special Elements Used in a Template Engine) and CVSS (CVSS 3.1: Score 10.0, Severity CRITICAL, Version 3.1, Vector String CVSS 3.1/AV:N/AC:L/PR:N/UI:N/S:C/H:H/A:H). A sidebar titled "On This Page" lists Required CVE Record Information, CNA: MITRE Corporation, CVE Program, and Authorized Data Publishers (CISA-ADP).

Figure 3 - CVE-2025-47916

Source: <https://www.cve.org/CVERecord?id=CVE-2025-47916>



National Vulnerability Database (NVD)

- Created: 2005
- Maintained by: National Institute of Standards and Technology (NIST)
- URL: <https://nvd.nist.gov/>

The NVD supplements CVE by providing enhanced metadata, including severity scores (CVSS), impact metrics, and fix information. It is a comprehensive repository of vulnerability management data used by government agencies, enterprises, and security tools to automate risk assessment and compliance. The NVD also publishes vulnerability feeds consumable by scanners and patch management systems.

The screenshot shows the NVD details page for CVE-2025-47916. At the top, there's a navigation bar with the NIST logo and a 'VULNERABILITIES' button. Below the header, the title 'CVE-2025-47916 Detail' is displayed. The main content area has two columns. The left column, titled 'Description', contains a detailed technical description of the vulnerability in Invision Community. The right column, titled 'QUICK INFO', lists the following details:

- CVE Dictionary Entry: CVE-2025-47916
- NVD Published Date: 05/16/2025
- NVD Last Modified: 06/20/2025
- Source: MITRE

Below the 'Description' section, there's a 'Metrics' section with tabs for CVSS Version 4.0, CVSS Version 3.x, and CVSS Version 2.0. It shows the following data:

Source	Base Score	Vector
NVD: NVD	9.8 CRITICAL	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
CNA: MITRE	10.0 CRITICAL	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

Figure 4 - NVD details for CVE-2025-47916

Source: <https://nvd.nist.gov/vuln/detail/CVE-2025-47916>



Common Vulnerability Scoring System (CVSS)

- Created: 2005
- Maintained by: FIRST (Forum of Incident Response and Security Teams)
- URL: <https://www.first.org/cvss/>

CVSS is a standardized framework to quantify the severity of vulnerabilities on a numeric scale from 0 (low) to 10 (critical). This scoring facilitates objective risk prioritization and consistent communication across teams.

Structure of the CVSS Score

- Base Metrics (Required)

These reflect the intrinsic properties of the vulnerability, independent of environment or time. This is the core of CVSS and always required.

Metric	Description	Values
Attack Vector (AV)	How the vulnerability is exploited	Network (N), Adjacent (A), Local (L), Physical (P)
Attack Complexity (AC)	Conditions beyond the attacker's control required?	Low (L), High (H)
Privileges Required (PR)	Level of privileges needed by the attacker	None (N), Low (L), High (H)
User Interaction (UI)	Is user interaction required?	None (N), Required (R)
Scope (S)	Can the exploit affect other system components?	Unchanged (U), Changed (C)
Confidentiality (C)	Loss of confidentiality?	None (N), Low (L), High (H)
Integrity (I)	Loss of data integrity?	None, Low, High
Availability (A)	Loss of availability?	None, Low, High



Base Score

Select values for all base metrics to generate score

Attack Vector (AV)	Scope (S)
Network (N) Adjacent (A) Local (L) Physical (P)	Unchanged (U) Changed (C)
Attack Complexity (AC)	Confidentiality (C)
Low (L) High (H)	None (N) Low (L) High (H)
Privileges Required (PR)	Integrity (I)
None (N) Low (L) High (H)	None (N) Low (L) High (H)
User Interaction (UI)	Availability (A)
None (N) Required (R)	None (N) Low (L) High (H)

Figure 5 - Base Score CVSS calculator

Source: <https://www.first.org/cvss/calculator/3-1>

- Temporal Metrics (Optional)

These account for factors that change over time, such as exploit availability and patch status.

Metric	Description
Exploit Code Maturity (E)	Is there an exploit in the wild?
Remediation Level (RL)	Is a fix available and what kind?
Report Confidence (RC)	How reliable is the vulnerability report?



Temporal Score

Exploit Code Maturity (E)

Not Defined (X) Unproven (U) Proof-of-Concept (P) Functional (F) High (H)

Remediation Level (RL)

Not Defined (X) Official Fix (O) Temporary Fix (T) Workaround (W) Unavailable (U)

Report Confidence (RC)

Not Defined (X) Unknown (U) Reasonable (R) Confirmed (C)

Select values for all base metrics to generate score

Figure 6 - Temporal Score CVSS calculator

Source: <https://www.first.org/cvss/calculator/3-1>

- Environmental Metrics (Optional)

Environmental Score

Confidentiality Requirement (CR)

Not Defined (X) Low (L) Medium (M) High (H)

Integrity Requirement (IR)

Not Defined (X) Low (L) Medium (M) High (H)

Availability Requirement (AR)

Not Defined (X) Low (L) Medium (M) High (H)

Modified Attack Vector (MAV)

Not Defined (X) Network Adjacent Network Local Physical

Modified Attack Complexity (MAC)

Not Defined (X) Low High

Modified Privileges Required (MPR)

Not Defined (X) None Low High

Modified User Interaction (MUI)

Not Defined (X) None Required

Modified Scope (MS)

Not Defined (X) Unchanged Changed

Modified Confidentiality (MC)

Not Defined (X) None Low High

Modified Integrity (MI)

Not Defined (X) None Low High

Modified Availability (MA)

Not Defined (X) None Low High

Select values for all base metrics to generate score

Figure 7 - Environmental Metrics CVSS calculator

Source: <https://www.first.org/cvss/calculator/3-1>



Score Interpretation

Score Range	Severity
0.0	None
0.1 – 3.9	Low
4.0 – 6.9	Medium
7.0 – 8.9	High
9.0 – 10.0	Critical

Version History

Version	Release Date	Key Features
CVSS v1	2005	Initial release. Adopted by NVD. Limited adoption due to design flaws and lack of flexibility.
CVSS v2	2007	Introduced Base, Temporal, and Environmental metric groups. Widely adopted (especially in the NVD and CVE ecosystem).
CVSS v3.0	June 2015	Improved granularity and accuracy. Added Scope , reworked privilege metrics. Better suited for modern threats.
CVSS v3.1	June 2019	Clarified metric definitions. Improved documentation and scoring guidance. Became the most used version to date.
CVSS v4.0	October 2023	Major overhaul. Introduced Supplemental Metrics and Exploitability, Automatable, and Recovery metrics. Designed to better align with EPSS and modern threat intelligence. Not yet adopted by all tools.



Key Limitations

- CVSS does not assess exploit activity (use EPSS or LEV for that)
- It does not account for exposure (e.g., vulnerability behind a firewall)
- The Base Score is generic and may not reflect actual risk in your context



Exploit Prediction Scoring System (EPSS)

- Created: 2020
- Maintained by: FIRST and community contributors
- URL: <https://www.first.org/epss/>

EPSS is a predictive model estimating the likelihood that a given vulnerability will be actively exploited within the next 30 days. By analyzing historical exploit trends, vulnerability attributes, and threat intelligence data, EPSS helps security teams anticipate and prioritize patches based not only on severity but on exploitation probability. EPSS complements CVSS by adding a temporal risk factor to vulnerability management.



The screenshot shows a table titled "Top rated CVEs from the last ninety days". The table lists 48 CVE entries, each with its name, a percentage value, and an EPSS score. The table has 6 columns and 8 rows of data. The data is as follows:

CVE-2025-47916 89.9%	CVE-2025-48827 68.0%	CVE-2024-51978 46.4%	CVE-2025-4322 30.8%	CVE-2025-5086 20.1%	CVE-2025-44840 16.1%
CVE-2025-49113 84.7%	CVE-2025-34028 64.7%	CVE-2025-35939 44.9%	CVE-2025-2010 27.6%	CVE-2025-32814 18.5%	CVE-2025-44841 16.1%
CVE-2025-47812 83.4%	CVE-2025-31324 63.8%	CVE-2024-46506 43.2%	CVE-2025-45985 27.4%	CVE-2025-30397 18.4%	CVE-2025-44842 16.1%
CVE-2025-4427 83.3%	CVE-2025-2011 61.7%	CVE-2025-50201 40.9%	CVE-2025-4428 25.3%	CVE-2025-42999 18.1%	CVE-2025-44844 16.1%
CVE-2025-27007 78.2%	CVE-2025-48828 60.9%	CVE-2024-51977 40.7%	CVE-2025-3486 25.0%	CVE-2025-33053 18.1%	CVE-2025-44845 16.1%
CVE-2025-34073 76.8%	CVE-2025-32433 58.4%	CVE-2025-49619 40.6%	CVE-2025-3884 22.3%	CVE-2025-47539 17.2%	CVE-2025-44848 16.1%
CVE-2024-48766 74.0%	CVE-2025-27920 57.6%	CVE-2025-49132 33.6%	CVE-2025-32815 22.2%	CVE-2024-13322 16.5%	CVE-2025-44860 16.1%
CVE-2025-32432 72.6%	CVE-2025-4632 48.7%	CVE-2025-4396 33.5%	CVE-2025-3935 20.1%	CVE-2025-44839 16.1%	CVE-2025-44863 16.1%

Source: https://first.org/epss/data_stats, 2025-07-16

Figure 8 - EPSS score for top rated CVE

Source: https://www.first.org/epss/data_stats



MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge)

- Created: 2013
- Maintained by: MITRE Corporation
- URL: <https://attack.mitre.org/>

ATT&CK is a knowledge base of adversary tactics and techniques derived from real-world cyber attacks. It describes the behavior and methods attackers use to infiltrate, move laterally, escalate privileges, and exfiltrate data. ATT&CK provides a common language for threat hunting, detection, and defense and helps security teams build effective detection rules and incident response strategies. Unlike CVE or CWE, ATT&CK focuses on attack behaviors rather than individual vulnerabilities. The ATT&CK framework is structured based on observations from real-world incidents and publicly available threat intelligence. It is updated continuously with contributions from the cybersecurity community.

MITRE ATT&CK is often mapped with:

- CVEs (which identify vulnerabilities)
- NIST frameworks
- Sigma, YARA, and OSQuery rules



← → ⌛ attack.mitre.org

MITRE | ATT&CK®

Matrices ▾ Tactics ▾ Techniques ▾ Defenses ▾ CTI ▾ Resources ▾ Benefactors ▾ Blog ↗ Search 🔎

ATT&CK Matrix for Enterprise

layout: side ▾ show sub-techniques hide sub-techniques

Reconnaissance	Resource Development	Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
10 techniques	8 techniques	11 techniques	16 techniques	23 techniques	14 techniques	45 techniques	17 techniques	33 techniques	9 techniques	17 techniques	18 techniques	9 techniques	15 techniques
Active Scanning (3)	Acquire Access	Content Injection	Cloud Administration Command	Account Manipulation (7)	Abuse Elevation Control Mechanism (6)	Adversary-in-the-Middle (4)	Exploitation of Remote Services	Adversary-in-the-Middle (4)	Application Layer Protocol (5)	Automated Exfiltration (1)	Account Access Removal		
Gather Victim Host Information (4)	Acquire Infrastructure (8)	Drive-by Compromise	Command and Scripting Interpreter (12)	BITS Jobs	Access Token Manipulation (5)	Brute Force (4)	Internal Spearphishing	Archive Collected Data (3)	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction (1)		
Gather Victim Identity Information (3)	Compromise Accounts (3)	Exploit Public-Facing Application	Container Administration Command	Boot or Logon Autostart Execution (14)	Account Manipulation (7)	Credentials from Password Stores (6)	Lateral Tool Transfer	Audio Capture	Content Injection	Exfiltration Over Alternative Protocol (3)	Data Encrypted for Impact		
Gather Victim Network Information (6)	Compromise Infrastructure (8)	External Remote Services	Deploy Container	Boot or Logon Initialization Scripts (5)	Build Image on Host	Exploitation for Credential Access	Remote Service Session Hijacking (2)	Automated Collection	Data Encoding (2)	Data Obfuscation (3)	Data Manipulation (3)		
Gather Victim Org Information (4)	Develop Capabilities (4)	Hardware Additions	ESXi Administration Command	Cloud Application Integration	Debugger Evasion	Forced Authentication	Remote Services (8)	Browser Session Hijacking	Dynamic Resolution (3)	Exfiltration Over C2 Channel	Defacement (2)		
Phishing for Information (4)	Establish Accounts (3)	Phishing (4)	Exploitation for Client Execution	Compromise Host Software Binary	Deobfuscate/Decode Files or Information	Forge Web Credentials (2)	Clipboard Data	Clipbaord Data	Encrypted Channel (2)	Exfiltration Over Other Network Medium (1)	Disk Wipe (2)		
Search Closed Sources (2)	Obtain Capabilities (7)	Replication Through Removable Media	Input Injection	Create or Modify System Process (5)	Deploy Container	Input Capture (4)	Data from Cloud Storage	Data from Configuration Repository (2)	Fallback Channels	Exfiltration Over Physical Medium (1)	Email Bombing		
Search Open Technical Databases (5)	Stage Capabilities (6)	Supply Chain Compromise (3)	Inter-Process Communication (3)	Create Account (3)	Direct Volume Access	Modify Authentication Process (9)	Software Deployment Tools	Data from Information Repositories (5)	Hide Infrastructure	Endpoint Denial of Service (4)	Financial Theft		
Search Open Websites/ Domains (3)	Trusted Relationships	Valid Accounts (4)	Native API	Create or Modify System Process (5)	Domain or Tenant Policy Modification (2)	Multi-Factor Authentication Interception	Taint Shared Content	Data from Local System	Ingress Tool Transfer	Exfiltration Over Web Service (4)	Firmware Corruption		
Search Victim-Owned Websites	Wi-Fi Networks	Scheduled Task/ Job (5)	Event Triggered Execution (17)	Escape to Host	Email Spoofing	Multi-Factor Authentication Request Generation	Debugger Evasion	Data from Network Shared Drive	Multi-Stage Channels	Inhibit System Recovery	Network Denial of Service (2)		
		Serverless Execution	Exclusive Control	Event Triggered Execution (17)	Execution Guardrails (2)	Network Sniffing	Device Driver Discovery	Data from Removable Media	Non-Application Layer Protocol	Non-Standard Port	Resource Hijacking (4)		
		Shared Modules	Exploitation for Privilege Escalation	Hijack Execution Flow (12)	File and Directory Permissions Modification (2)	OS Credential Dumping (8)	Domain Trust Discovery	Data Staged (2)	Protocol Tunneling	Protocol Tunneling	Service Stop		
		Software Deployment Tools	Hijack Execution Flow (12)	Implant Internal Image	Hide Artifacts (14)	Steal Application Access Token	File and Directory Discovery	Email Collection (3)	Proxy (4)	Remote Access Tools (3)	System Shutdown/ Reboot		
		System Services (3)	System Services (3)	Process Injection (12)	Hijack Execution Flow (12)	Steal or Forge Authentication Certificates	Group Policy Discovery	Input Capture (4)	Screen Capture	Traffic			
		User Execution (4)	User Execution (4)	Scheduled Task/ Job (5)	Impair Defenses (11)	Impersonation	Log						
		Windows Management Instrumentation	Windows Management Instrumentation	Valid Accounts (4)									
		Modify Registry	Modify Registry										

Figure 9 - MITRE ATT&CK main website page

Source: <https://attack.mitre.org/>



MITRE D3FEND

- Created: 2021
- Maintained by: MITRE Corporation
- URL: <https://d3fend.mitre.org/>

D3FEND is a knowledge base of cybersecurity defensive techniques and countermeasures, designed as a complement to ATT&CK. It catalogs methods defenders can implement to protect systems against attacker techniques, such as sandboxing, privilege separation, or encryption. D3FEND assists security architects in designing robust defense strategies grounded in standardized concepts.

Model	Harden						Detect						Isolate							
	Agent Authentication	Application Hardening	Credential Hardening	Message Hardening	Platform Hardening	Source Code Hardening	File Analysis	Identifier Analysis	Message Analysis	Network Traffic Analysis	Platform Monitoring	Process Analysis	User Behavior Analysis	Access Mediation	Access Policy Administration	Content Filtering	Execution Isolation	Network Isolation	Deco Environment	
Biometric Authentication	Biometric Authentication	Application Configuration Hardening	Certificate Pinning	Message Authentication	Bootloader Authentication	Credential Scrubbing	Dynamic Analysis	Homoglyph Detection	Sender MTA Reputation Analysis	Administrative Network Activity Analysis	File Integrity Monitoring	Database Query String Analysis	Authentication Event Thresholding	Credential Transmission Scoping	Domain Trust Policy	Content Modification	Application-based Process Isolation	Broadcast Domain Isolation	Connection Honey:	
	Certificate-based Authentication	Credential-based Elimination	Credential Rotation	Message Encryption	Disk Encryption	Integer Range Validation	Emulated File Analysis	Identifier Activity Analysis	Sender Reputation Analysis	Firmware Behavior Analysis	File Access Pattern Analysis	Authorization Event Thresholding	IO Port Restriction	Local File Permissions	Content Excision	Executable Allowlisting	DNS Allowlisting	Integral Honey:	Standby Honey:	
	Multi-factor Authentication	Exception Handler Pointer Validation	Password Rotation	Transfer Agent Authentication	Driver Load Integrity Checking	Pointer Validation	File Content Analysis	Identifier Reputation Analysis	File Sequence Emulation	Firmware Embedded Monitoring Code	Indirect Branch Call Analysis	Credential Compromise Scope Analysis	Network Access Mediation	User Account Permissions	Content Format Conversion	Executable Denylisting	DNS Denylisting	Forward Resolution Domains Denylisting	Hierarchical Domain Denylisting	
	Password Authentication	Pointer Authentication	Strong Password Policy	One-time Password	Memory Block Start Validation	File Content Rules	Domain Name Reputation Analysis	Certificate Analysis	Active Certificate Analysis	Firmware Verification	Process Code Segment Verification	Domain Account Monitoring	LAN Access Mediation	Content Rebuild	Content Substitution	Content Quarantine	Kernel-based Process Isolation	Forward Resolution IP Denylisting	Homoglyph Denylisting	Forward Resolution IP Denylisting
	Token-based Authentication	Process Segment Execution Prevention	Change Default Password	Hardware-based Write Protection	Null Pointer Checking	File Hashing	File Hash Reputation Analysis	Passive Certificate Analysis	Peripheral Firmware Verification	Process Self-modification Detection	Job Function Access Pattern Analysis	Network Resource Access Mediation	Content Validation	Content Quarantine	Content Substitution	Content Rebuild	Forward Resolution Domains Denylisting	Reverse Resolution IP Denylisting	Encrypted Tunnels	
		Segment Address Offset Randomization		RF Shielding	Software Update	Reference Nullification	IP Reputation Analysis	Client-server Payload Profiling	System Firmware Verification	Local Account Monitoring	Remote File Access Mediation	File Format Verification	File Content Decompression Checking	File Internal Structure Verification	File Metadata Consistency Validation	File Metadata Value Verification	File Magic Byte Verification	Network Traffic Filtering	Inbound Traffic Filtering	
		Stack Frame Canary Validation		System Configuration Permissions	Trusted Library	Variable Initialization	URL Reputation Analysis	Connection Attempt Analysis	Operating Mode Monitoring	Process Spawn Analysis	Resource Access Pattern Analysis	Web Session Access Mediation	File Metadata Consistency Validation	File Metadata Value Verification	File Internal Structure Verification	File Metadata Consistency Validation	File Magic Byte Verification	Network Traffic Filtering	Email Filtering	
				TPM Boot Integrity	Variable Type Validation		URL Analysis	DNS Traffic Analysis	Operating System Monitoring	Process Lineage Analysis	Script Execution Analysis	Session Duration Analysis	Endpoint-based Web Server Access Mediation	File Metadata Consistency Validation	File Metadata Value Verification	File Internal Structure Verification	File Metadata Consistency Validation	File Magic Byte Verification	Outbound Traffic Filtering	
								IPC Traffic Analysis	Input Device Analysis	System Call Analysis	User Geolocation Logon Pattern Analysis	Proxy-based Web Server Access Mediation	File Creation Analysis	File Session Activity Analysis	Operating Mode Restriction	Physical Access				
								Network Traffic Community Deviation	Memory Boundary Tracking	Shadow Stack Comparisons	User Data Transfer Analysis									

Figure 10 - MITRE D3FEND main website page

Source: <https://d3fend.mitre.org/>



CPE (Common Platform Enumeration)

- Created: 2006
- Maintained by: MITRE Corporation
- URL: <https://nvd.nist.gov/products/cpe>

CPE is a standardized method for naming and identifying classes of applications, operating systems, and hardware devices. It enables consistent referencing of products across various security tools and databases.

A CPE name uniquely identifies a product using a structured URI format:

```
cpe:2.3:<part>:<vendor>:<product>:<version>:<update>:<edition>:<language>:<sw_edition>:<target_sw>:<target_hw>:<other>
```

Example:

```
cpe:2.3:a:apache:http_server:2.4.51:.*:.*:.*:.*:.*:*
```

This entry would refer to Apache HTTP Server version 2.4.51.



← → ⌂ nvd.nist.gov/products/cpe/detail/FAFA958E-13B7-4718-B142-75C5F89F771F ⌂ ⌂ ⌂ ⌂

[PRODUCTS](#) [CPE](#)

CPE Summary

[Return to Search Listing](#)

CPE Names

Version 2.3: [cpe:2.3:a:apache:http_server:2.4.51:.*:.*:.*:.*:.*](#)

Version 2.2: [cpe:/a:apache:http_server:2.4.51](#)

[Read information about CPE Name encoding](#)

 **CPE NAME COMPONENTS** SELECT A COMPONENT TO SEARCH FOR SIMILAR CPES

Part: a
Vendor: apache
Product: http_server
Version: 2.4.51

Metadata

QUICK INFO

Created On: 10/20/2021
Last Modified On: 10/20/2021

Figure 11 - CPE détails for apache server 2.4.51

Source: <https://nvd.nist.gov/products/cpe/detail/FAFA958E-13B7-4718-B142-75C5F89F771F>



Security Content Automation Protocol (SCAP)

- Created: 2007
- Maintained by: National Institute of Standards and Technology (NIST)
- URL: <https://csrc.nist.gov/projects/security-content-automation-protocol>

SCAP is a suite of open standards that enable automated vulnerability management, policy compliance, and security measurement. It integrates various specifications such as CVE, CPE (Common Platform Enumeration), and CVSS. SCAP allows security tools to exchange standardized data formats for automated scanning, assessment, and reporting, improving interoperability between scanners and security products.

OpenSCAP – Open Security Content Automation Protocol

OpenSCAP is an open-source framework designed to assess and enforce system security and compliance. It provides a free implementation of the SCAP (Security Content Automation Protocol) standards. OpenSCAP enables system administrators and security professionals to automate security audits and verify that systems conform to established security baselines.

Documentation: <https://github.com/OpenSCAP/openscap>

- Key Features
 - *Compliance auditing*: Evaluate systems against benchmarks like CIS, PCI-DSS, DISA STIGs.
 - *Vulnerability scanning*: Detect known issues based on CVEs and OVAL definitions.
 - *Security baselining*: Validate configurations across multiple hosts to ensure standardization.
 - *Remediation scripts*: Generate automated fixes for non-compliant systems.



I.2. Automated vulnerability assessment – OpenVAS

Vulnerability assessment is the process of identifying, classifying, and prioritizing security weaknesses in systems, applications, and networks. It forms a critical part of a defense-in-depth strategy, allowing administrators to detect misconfigurations and software vulnerabilities before attackers can exploit them. However, *it's important to note that vulnerability assessments do not replace penetration testing*. While vulnerability assessments focus on identifying and categorizing potential vulnerabilities, penetration testing involves actively exploiting those vulnerabilities to understand the real-world impact and effectiveness of security measures. Both are essential components of a comprehensive security strategy.

OpenVAS (Open Vulnerability Assessment Scanner) is a powerful, free, and open-source vulnerability scanning framework. It is part of the Greenbone Vulnerability Management (GVM) suite and provides automated scanning, reporting, and compliance tools suitable for enterprise environments.

Documentation:

- <https://www.openvas.org/>
- <https://github.com/greenbone/>

How OpenVAS Works

OpenVAS uses a database of Network Vulnerability Tests (NVTs) written in NASL (Nessus Attack Scripting Language). The scanner performs a variety of checks including:

- Port scanning (TCP/UDP)
- Service and version detection
- Operating system fingerprinting
- Web application vulnerability checks
- CVE-based software vulnerability scans
- Credentialed and non-credentialed scanning
- The scan results are scored using the CVSS (Common Vulnerability Scoring System) and classified by severity (Low, Medium, High, Critical).



Architecture Overview

OpenVAS typically includes the following components:

- OpenVAS Scanner – the engine that performs vulnerability scans
- GVMd (Greenbone Vulnerability Manager daemon) – manages scan tasks and user permissions
- gvmd-cli / GSA (Greenbone Security Assistant) – web and CLI interfaces for managing scans and results
- PostgreSQL database – stores scan results and configurations
- NVT Feed – updated regularly with vulnerability test scripts from Greenbone

Typical Use Cases

- Internal network scanning to identify vulnerable machines and services
- Audit of exposed services on perimeter firewalls and DMZ hosts
- Web application assessments for known CVEs
- Compliance checks (e.g., PCI-DSS, ISO 27001, CIS benchmarks)
- Integration into CI/CD pipelines for DevSecOps (with CLI or API)



Screenshots

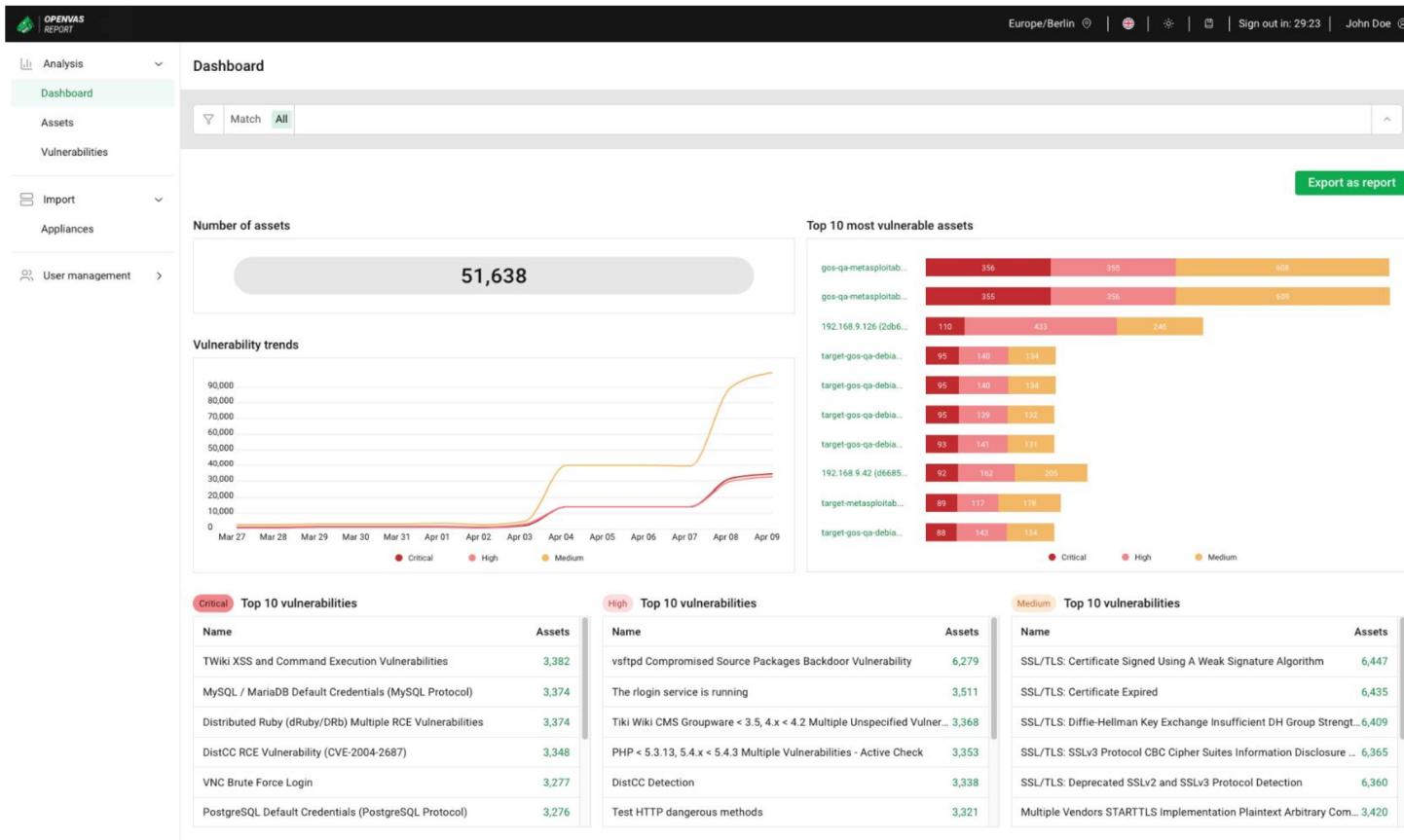


Figure 12 - OpenVAS dashboard

Source: https://docs.greenbone.net/OPENVAS-REPORT-Manual/en/_images/dashboard.png



OPENVAS REPORT

Europe/Berlin | Sign out in: 29:58 | John Doe

Assets

Match All

Host name	IP address	Operating system	Appliance name	Last scan	Asset ID	Criticality	Tags	Action
TSMIM4BD87BU	15.192.206.234	Linux/Unix	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 09:43:10 AM	0c025287-ce65-4485-8a94-bd188066db3a			
3LU57EEIJ65Y	155.17.194.168	Linux/Unix	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 10:03:49 AM	f70fe1c1-4101-4474-a3e1-d215bbb78749			
AJ0DDCZIJSWL	104.214.99.194	Debian GNU/Linux	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 09:48:34 AM	e0f7c230-4473-4333-a3ae-ec6211ad7c36			
129992H53KPKW	7.236.12.237	Debian GNU/Linux	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 09:58:56 AM	1f3b4e9b-ab53-474e-928e-4d490c7c2c94			
OBHS1HYGYCS6	183.48.0.176	Linux/Unix	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 09:54:57 AM	9e67f3d4-e14d-484b-9dcf-b9839c588efa			
XCT34A1H2OCE	108.155.211.3	Ubuntu 16.04	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 10:03:49 AM	05c565d1-1f52-4608-a960-7f331f1be87c			
9KZFM1I7TXPK	133.114.245.96	Ubuntu 16.04	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 09:48:34 AM	e581f253-e7e5-4a7a-b5c2-0c610e273e29			
OYIXB30EPTZU	187.226.251.51	Linux/Unix	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 10:03:49 AM	2f92f573-f0ec-445e-b9ff-26d9cb72bc0a			
SZ07SU5T7IC6	98.4.31.59	Greenbone OS (GOS)	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 10:03:49 AM	3b4b2329-b44e-47ea-9bd9-1474ac29168e			
KK9U00SASDBK	30.239.201.96	Greenbone OS (GOS)	QM appliance - GEA 6500 R2 - GOS 24.10.2...	Apr 08, 2025, 10:03:49 AM	1575648b-8e0c-440d-9d2e-c48c3ae75dd2			

Items per page: 10 1-10 of 4,983 items

< 1 2 3 4 5 ... 499 >

Figure 13 - OpenVAS Asset details

Source: https://docs.greenbone.net/OPENVAS-REPORT-Manual/en/_images/asset-details.png



Vulnerabilities

Severity ↑	EPSS score [%] ↑	EPSS percentile ↑	Vulnerability name	QoD ↑	Affected assets ↓	Solution type
6.0	89.1	99th	Samba 3.0.0 <= 3.0.25rc3 MS-RPC Remote Shell Command Execution Vulnerability - Active Check	75	3374	Vendor fix
5.0	79.9	99th	Tiki Wiki CMS Groupware < 12.11, 13.x < 15.4 Local File Inclusion Vulnerability	75	3394	Vendor fix
10 🔎	76.5	99th	TWiki XSS and Command Execution Vulnerabilities	75	3399	Vendor fix
4.3	75.2	99th	Weak Encryption Algorithm(s) Supported (SSH)	75	3374	Mitigation
9.0	69.5	99th	MySQL / MariaDB Default Credentials (MySQL Protocol)	75	3402	Mitigation
10	46.6	98th	Distributed Ruby (dRuby/DRb) Multiple RCE Vulnerabilities	75	3378	Mitigation
6.8 🔎	17.2	95th	SSL/TLS: OpenSSL CCS Man in the Middle Security Bypass Vulnerability	75	3357	Vendor fix
7.5	9.0	92nd	Tiki Wiki CMS Groupware < 3.5, 4.x < 4.2 Multiple Unspecified Vulnerabilities	75	3389	Vendor fix
6.4	8.0	92nd	Anonymous FTP Login Reporting	75	3363	Mitigation
4.3	6.6	91st	Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability	75	3362	Vendor fix

Items per page: 10 ⏮ 11–20 of 3,522 items

Figure 14 - OpenVAS vulnerabilities page

Source: https://docs.greenbone.net/OPENVAS-REPORT-Manual/en/_images/vulnerabilities-overview.png



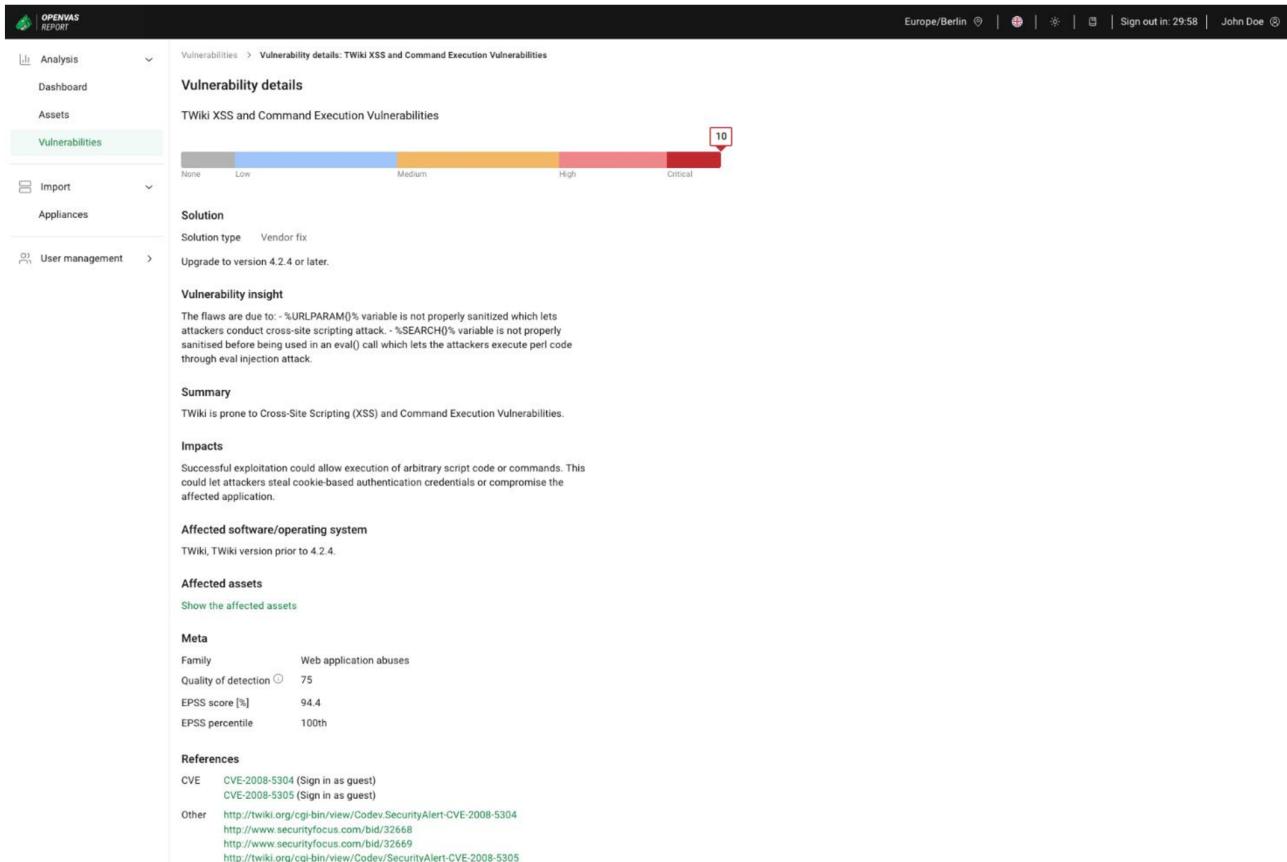


Figure 15 - OpenVAS vulnerability details

Source: https://docs.greenbone.net/OPENVAS-REPORT-Manual/en/_images/vulnerability-details.png



I.3. Automated vulnerability assessment – Nuclei

Nuclei is a fast and flexible open-source vulnerability scanner developed by ProjectDiscovery. It allows security professionals and system administrators to automatically detect vulnerabilities using YAML-based templates, making it highly customizable and scriptable for integration into CI/CD pipelines, asset discovery or routine assessments.

Unlike traditional scanners, Nuclei is template-driven and focuses on speed, modularity, and low false-positive rates.

Documentation:

- <https://github.com/projectdiscovery/nuclei>
- <https://projectdiscovery.io/nuclei>

Important Directories and Files

- `~/.local/nuclei-templates/`: Default location for downloaded templates
- `~/.config/nuclei/`: Stores metadata like update status, cache

Core Concept: Templates

At the heart of Nuclei lies the template system. Template structure:

- A unique ID for the template
- Essential information and metadata relevant to the template
- The designated protocol, such as HTTP, DNS, File, etc.
- Details specific to the chosen protocol, like the requests made in the HTTP protocol
- A series of matchers to ascertain the presence of findings
- Necessary extractors for data retrieval from the results



```
$ acrapx >> cat CVE-2025-0107.yaml
id: CVE-2025-0107

info:
  name: Palo Alto Networks Expedition - OS Command Injection
  author: iamnoob,pdresearch
  severity: critical
  description: |
    An OS command injection vulnerability in Palo Alto Networks Expedition enables an unauthenticated attacker to run arbitrary OS commands as the www-data user in Expedition, which results in the
    words, device configurations, and device API keys for firewalls running PAN-OS software.
  reference:
    - https://security.paloaltonetworks.com/PAN-SA-2025-0001
    - https://ssd-disclosure.com/ssd-advisory-palo-alto-expedition-rce-regionsdiscovery/
    - https://nvd.nist.gov/vuln/detail/CVE-2025-0107
classification:
  cvss-metrics: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
  cvss-score: 9.8
  epss-score: 0.22791
  epss-percentile: 0.95559
metadata:
  verified: true
  max-request: 1
  shodan-query: title:"Expedition"
  fofa-query: title=="Expedition Project"
tags: cve,cve2025,rce,paloalto,expedition

http:
  - raw:
    - |
      GET /API/regionsDiscovery.php?master=spark%3A%2F%2F{{interactsh_url}}:443&mask=26&project=your_project&devices=device1%2Cdevice2&mtserver=127.0.0.1%3A3306&mtuser=root&mtpassword=paloalto&t
      arquetPath=%2Ftmp&timezone=Europe%2FHelsinki&mlserver=127.0.0.1&debug=false&initDate=2023-01-01&endDate=2023-01-31 HTTP/1.1
      Host: {{Hostname}}
    matchers-condition: and
    matchers:
      - type: word
        part: body
        words:
          - 'msg":"Started'
          - '"success":true'
        condition: and
      - type: word
        part: interactsh_protocol
        words:
          - "dns"
# digest: 4a0a00473045022052d4c16374943beb476a248bc8f298132222dhcc5c7378cacc96abdd87a13c7202210094f8be4d4533a0c6989e8b589337d6a6e34e17104d5f31f336eaf9994984899f:922c64590222798bb761d5b6d8e72950
```

Figure 16 - Nuclei template example

Nuclei allows you to write your own templates to scan for internal vulnerabilities or undocumented services specific to your environment.



Nuclei installation

```
#First install go  
go install -v github.com/projectdiscovery/nuclei/v3/cmd/nuclei@latest  
cp /root/go/bin/nuclei /usr/local/go/bin/
```

Update Nuclei binary

Keeping Nuclei and its templates up to date is essential to ensure accurate and comprehensive vulnerability scanning.

```
nuclei -update # Updates the Nuclei engine (binary) to the latest version.
```

Incremental template update

```
nuclei -ut # Incrementally updates only the changed or new templates from the official repository.
```

Full reset of templates (hard refresh)

```
# Deletes the current templates and clones a fresh copy of the official repository.  
rm -rf ~/nuclei-templates/ 2>/dev/null  
git clone https://github.com/projectdiscovery/nuclei-templates ~/nuclei-templates/
```



Common Nuclei Usage Scenarios

The nuclei CLI tool provides flexible scanning capabilities across various targets and vulnerability categories. Below are typical usage patterns to perform CVE scans, target internal assets, apply custom templates, filter by severity, or control output format and rate:

```
# Scan a single target URL using all templates from default location
nuclei -u https://example.com

# Scan a single target URL using all CVE templates
nuclei -u https://example.com -t cves/

# Use custom internal templates on a target inside the local infrastructure
nuclei -u https://internal.app.local -t /opt/nuclei/custom-templates/

# Scan a list of targets for high and critical severity vulnerabilities
nuclei -list targets.txt -t vulnerabilities/ -severity critical,high

# Detect information exposure vulnerabilities and save output in JSON
nuclei -u https://example.com -t exposures/ -json -o exposures.json

# Perform DNS-related checks on multiple domains with a request rate limit
nuclei -list domains.txt -t dns/ -rate-limit 25
```



Nuclei scan example

Figure 17 - Nuclei scan example on <https://google.fr>



I.4. Linux Package Manager

Package managers are central to Linux system maintenance. They automate the installation, upgrade, configuration, and removal of software. However, as they operate with elevated privileges, they are critical to attack surfaces. This section focuses on the apt package manager used in Debian-based systems and explains how it ensures software integrity and authenticity.

APT Source Files: Format and Parameters

APT relies on repository definitions to locate and authenticate packages. These definitions are stored in source files which describe the repository URI, distribution suite, component sections, architectures, and associated GPG keys.

APT reads its software source configuration from the following locations:

- `/etc/apt/sources.list`: Main file listing repository URIs.
- `/etc/apt/sources.list.d/*.list`: Supplementary list files for modular configuration.

There are two supported file formats:

- Legacy `.list` format (single-line entries)
- Modern `.sources` format (key-value entries)

Legacy .list format example

```
deb http://deb.debian.org/debian bookworm main contrib non-free
deb-src http://deb.debian.org/debian bookworm main contrib non-free
```



Modern .sources format

These files follow a *key: value* format and allow fine-grained configuration of repository properties.

- Types

Specifies the type of packages to retrieve:

- deb – Binary packages (default for installation)
- deb-src – Source code packages (used for audit or recompilation)

- URIs

Defines the location of the repository:

- Supports http://, https://, ftp://, file://, cdrom:
- Multiple URIs can be listed

- Suites

Indicates the distribution release:

- Named versions: bookworm, jammy, etc.
- Aliases: stable, testing, unstable
- Special: *-security, *-updates, *-backports

- Components

Defines package categories:

- Debian: main, contrib, non-free, non-free-firmware
- Ubuntu: main, universe, restricted, multiverse

- Architectures

Specifies target CPU architectures:

- Common values: amd64, i386, arm64, riscv64



- Signed-By

Specifies the GPG key used to verify the repository signature:

- Replaces global apt-key
- Increases repository isolation and trust control

Avoid legacy apt-key. Always use Signed-By for key scoping.

Modern .source format example

```
# /etc/apt/sources.list.d/ubuntu.sources

Types: deb
URIs: http://archive.ubuntu.com/ubuntu
Suites: noble noble-updates noble-backports
Components: main restricted universe multiverse
Signed-By: /usr/share/keyrings/ubuntu-archive-keyring.gpg

Types: deb
URIs: http://security.ubuntu.com/ubuntu
Suites: noble-security
Components: main restricted universe multiverse
Signed-By: /usr/share/keyrings/ubuntu-archive-keyring.gpg
```

Integrity and Authenticity of Linux Software Packages

Ensuring the integrity and authenticity of packages is a core security feature of modern Linux distributions. APT uses digital signatures based on PGP (see: Erreur ! Source du renvoi introuvable.) to verify that packages have not been tampered with and originate from trusted sources.



Principle of Package Signing

Each package or repository is signed by its maintainer using a PGP private key. The verification process works as follows:

- Signing:
 - The package maintainer generates a digital signature using a private key.
 - This signature is based on a hash of the package content.
- Verification:
 - The user's system stores the corresponding public key.
 - When a package is downloaded, APT verifies the signature using this key.

If the computed hash matches the one from the signature, the package is considered valid.

This process ensures:

- The package has not been altered (integrity)
- The package comes from a trusted source (authenticity)

Key Locations and Trust

- [*/etc/apt/trusted.gpg.d/*](#): Legacy location for trusted public keys (deprecated)
- [*/usr/share/keyrings/*](#) or [*/etc/apt/keyrings/*](#): Recommended for scoped GPG keys used with Signed-By

The use of */etc/apt/trusted.gpg.d/* is discouraged due to security risks associated with its automatic trust of any added keys, potentially leading to the installation of malicious software. Instead, */usr/share/keyrings/* or */etc/apt/keyrings/* are recommended as they allow for better key management and explicit trust associations with specific repositories, enhancing security and maintainability.



Key Management and *signed-by*

Important: *The legacy tool apt-key is deprecated.*

Instead of trusting all keys system-wide, modern best practice is to use the *signed-by* directive in *.sources* files.

This binds a specific repository to a specific GPG key, preventing cross-repo trust issues.

- Example:

```
Signed-By: /usr/share/keyrings/ubuntu-archive-keyring.gpg
```

Security Implications

If signature verification fails:

- The package will not be installed
- APT will return an error and abort the operation

This prevents:

- Man-in-the-middle attacks
- Malicious repository injection
- Compromised package deployment

APT Tools: apt, apt-cache, and apt-file

APT (Advanced Package Tool) provides several command-line tools to interact with Debian-based package systems. Each has a specific role in managing, querying, and exploring packages.



apt: The High-Level Package Manager

Introduced as a user-friendly frontend, apt combines functionalities from apt-get and apt-cache.

- Main uses:

- Installing, updating, and removing packages
- Refreshing the package index
- Displaying useful information

```
apt update                      # Refresh package lists
apt upgrade                     # Upgrade all installed packages
apt full-upgrade                # Upgrade, including packages requiring dependency changes
apt install nmap                  # Install a package
apt remove apache2               # Uninstall a package (keep config files)
apt purge apache2                # Uninstall and remove configuration files
apt list --installed             # List installed packages
apt show openssh-server          # Show package details
```

apt-cache: Package Metadata Lookup

apt-cache allows detailed exploration of the local package index without modifying the system. It is useful for searching and inspecting metadata:

- Dependencies
- Reverse dependencies
- Package versions and descriptions

```
apt-cache search firewall          # Search for packages by keyword
apt-cache show ufw                 # Show metadata for a specific package
apt-cache depends nginx            # List package dependencies
apt-cache rdepends python3         # List reverse dependencies
```



apt-file: Search Contents of Packages

apt-file allows you to search for files contained within packages, even if those packages are not installed. It is mainly used for finding which package provides a specific command or configuration file. It is useful to verify that a file won't be deleted or modified during an update.

```
apt-file update      # Update its internal database  
apt-file search zlib.so # Find the package providing this binary  
apt-file list nginx    # List all files provided by nginx
```



I.5. Centralized Patch & Endpoint Management

In a professional environment, centralized patch and endpoint management systems are essential for maintaining system integrity, security compliance, and operational continuity across all devices. While these two responsibilities, patch management and endpoint management, can be handled by separate tools, some platforms combine both capabilities into a unified solution. These tools allow administrators to remotely manage software updates, deploy configurations, and monitor endpoints from a unified console.

ManageEngine Endpoint Central

ManageEngine Endpoint Central is a unified endpoint management (UEM) solution developed by Zoho Corporation. It provides a centralized platform for managing, securing, and auditing workstations and servers, whether Linux, Windows, or macOS, from a single web interface.

Although originally focused on Windows fleets, Endpoint Central has evolved to support Linux systems, offering key functionalities for patching, asset tracking, and remote administration.

Documentation: <https://www.manageengine.com/fr/desktop-central/>

Key Features for Linux Systems

- Patch Management
 - Automatically scans for missing patches and applies OS-level updates using native tools (apt, yum, dnf, etc.).
 - Schedule patch deployment by group or device
 - Define test and production deployment rings
 - Track patch success/failure from a dashboard
- Software Inventory and License Tracking
 - Maintains a detailed software inventory of installed packages and versions, useful for compliance audits and license management.
- Remote Control and Shell Access
 - Offers secure remote terminal sessions to Linux servers and desktops for troubleshooting or configuration tasks.
- Custom Script Deployment
 - Supports pushing and executing shell scripts on selected devices or groups, making it ideal for configuration management tasks.
- Reporting and Compliance Dashboards



- Generate reports for:
 - Systems missing critical updates
 - Patch deployment status
 - System health and compliance metrics

Architecture Overview

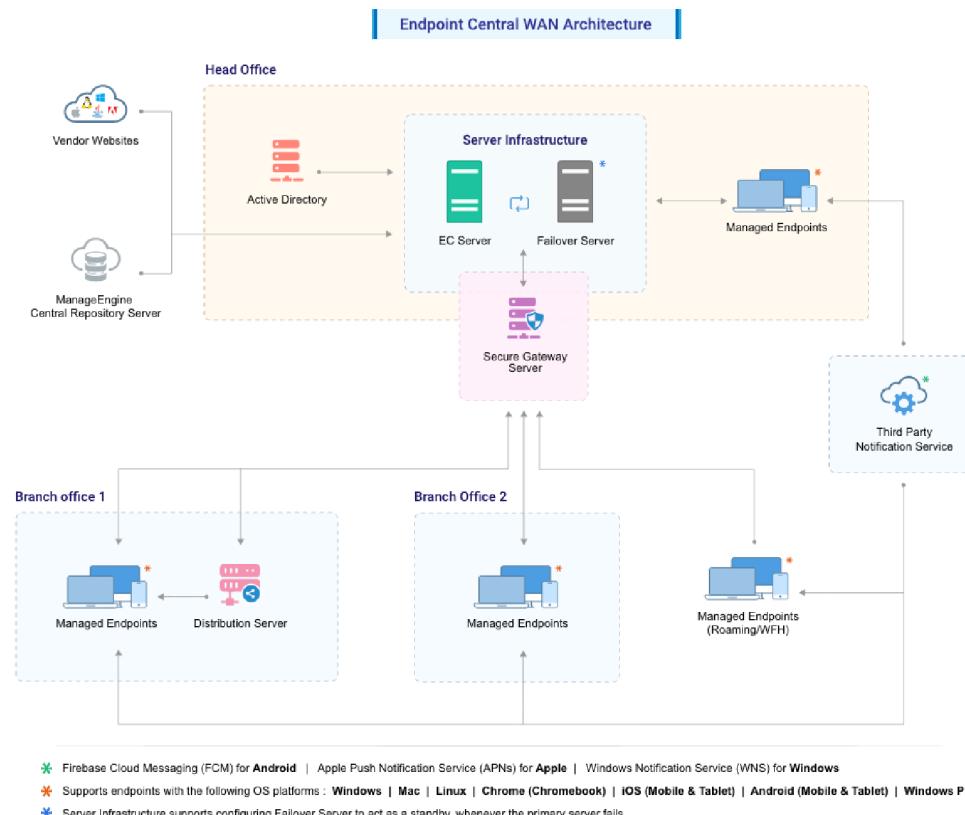


Figure 18 - Endpoint Centrale architecture overview

Source : <https://www.manageengine.com/products/desktop-central/images/endpoint-central-wan-architecture.png>



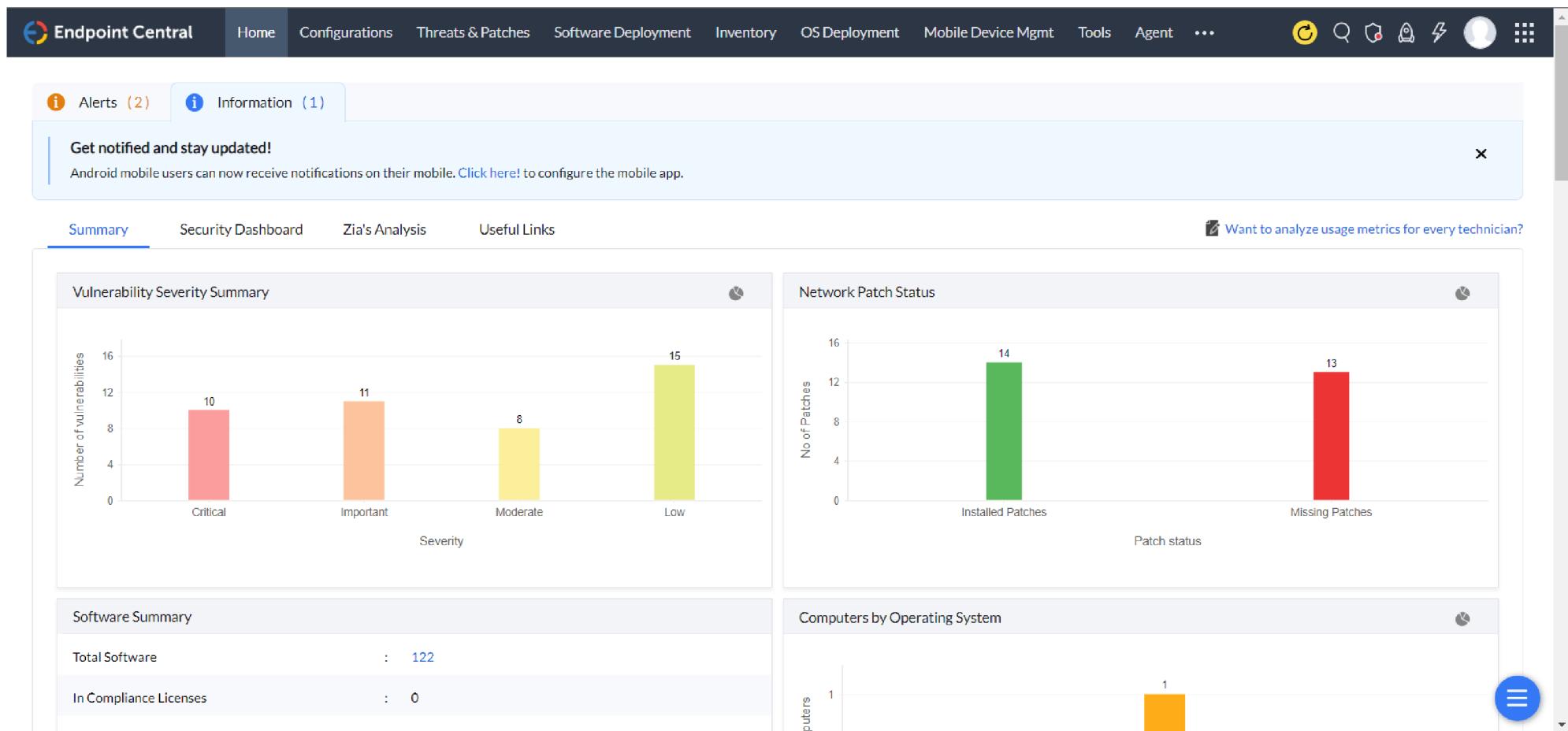


Figure 19 - Endpoint Central interface

Source: <https://www.manageengine.com/products/desktop-central/help/images/ec-console-3.png>



Landscape: Ubuntu Systems Management at Scale

Landscape is Canonical's centralized management tool designed to administer, monitor, and automate tasks across large fleets of Ubuntu machines. It is particularly suited for environments that require centralized package management, compliance, and reporting, whether on-premises, in the cloud, or in hybrid infrastructures.

Documentation: <https://ubuntu.com/landscape>

Key Features

- Patch & Package Management
 - Landscape lets administrators view, schedule, and apply updates and security patches across all registered Ubuntu systems.
 - Approve or reject updates manually or automatically
 - Group systems by role or environment (e.g., dev, prod)
 - Schedule updates outside business hours
- System Inventory and Monitoring
 - Hardware specs and software inventory
 - Available updates and upgrade history
 - System performance (CPU, memory, disk usage)
- User and Role Management
 - Supports role-based access control (RBAC) for delegating management responsibilities across teams.
- Script Automation
 - Enables batch execution of custom shell scripts across selected machines, useful for enforcing configuration consistency.
- Compliance & Reporting
 - Landscape allows you to generate reports on package status, update history, and compliance posture for audits.



Landscape

ORGANISATION
Onward, Inc.

- Overview
- Instances
- Activities
- Scripts
- Profiles >
- Repositories >
 - Mirrors
 - Repository profiles
 - GPG Keys
 - APT Sources
- Monitoring >
- Org. settings >
- Help >

John Smith

Notifications

Log out

Overview

Upgrades

Up to date	5070 instances
Regular	5915 instances
Security	4225 instances



Offline
145 instances
ESM updates are disabled
18 instances

Online
10840 instances
Duplicate
0 instances

Reboot required
524 instances

Upgrades available

Instances	Packages	USNs
nginx.lhr.canonical.com	13	
nginx.iad.canonical.com	10	
nginx.ewr.canonical.com	7	
Jane Doe Laptop	7	
John Doe Desktop	9	
Staging Web Server	6	
dev-web-server.lxd	333	

Activities

Requires approval	In progress

All activities have been approved
There are currently no pending approval requests. Check back later for any new approval activities

Figure 20 - Landscape interface

Source: <https://canonical.com/blog/canonical-releases-landscape-24-04-lts>



II. Further Resources and Recommended Readings

Web site

- <https://wiki.ubuntu.com/Security/Features>
- <https://cyber.gouv.fr/publications/recommandations-de-securite-relatives-un-systeme-gnulinux>
- <https://blog.stephane-robert.info/>
- <https://downloads.cisecurity.org/>
- <https://www.it-connect.fr/cours-tutoriels/administration-systemes/linux/>
- <https://github.com/imthenachoman/How-To-Secure-A-Linux-Server>
- <https://www.linuxfoundation.org/lf-security>

Playground and labs

- <https://webvm.io/>
- <https://labex.io/projects/category/linux>



Books

- # Ward, B. (2021). How Linux Works, 3rd Edition: What Every Superuser Should Know. San Francisco, CA: No Starch Press.

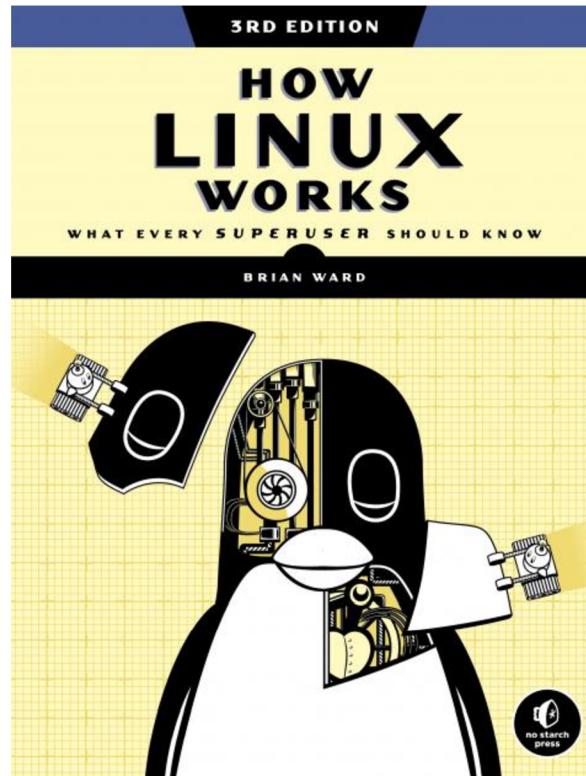


Figure 21 - How Linux Works, 3rd Edition: What Every Superuser Should Know

Source: <https://nostarch.com/howlinuxworks3>

