Proceedings of the
13th International Conference on Applied Informatics
Eger, Hungary, February 19–21, 2026
DOI: 10.17048/icai.2026.161
pp. 161–168

Comparative Analysis of Vulnerability Dynamics in Operating Systems and Containerized Environments

Ferenc Koczka

Eszterházy Károly Catholic University
ferenc.koczka@uni-eszterhazy.hu

Abstract. One of the most significant challenges in modern IT infrastructure management is maintaining the delicate equilibrium between technical debt and security integrity. In environments characterized by rapidly evolving requirements, research and economic priorities often necessitate the deployment of ad-hoc solutions. This process typically results in significant heterogeneity, where legacy systems and sub-optimally configured endpoints coexist. The central hypothesis of this research is that the "freshly installed" state of an operating system – even with a minimal package selection – does not, in itself, constitute a guarantee of security. This study performs a comparative analysis of the security profiles of traditional, monolithic operating system installations versus modern, Proxmox-based containerized solutions, with specific regard to the vulnerability lifecycle and the recommendations of the CIS (Center for Internet Security) and NIST (National Institute of Standards and Technology).

Keywords: vulnerability management, containerization, operating systems, security drift, CIS benchmarks, NIST SP 800-53, NIST SP 800-190, CVSS, SIEM, Wazuh, Linux containers, attack surface reduction, cybersecurity hardening

1. Theoretical Framework and Regulatory Environment

The theoretical foundation of this study is established upon international cybersecurity frameworks that define protection levels through quantifiable and auditable

parameters. Together, these standards constitute the reference framework against which a system's "security drift" – the measurable variance between the required security baseline and the actual operational state – can be evaluated.

1.1. NIST SP 800-53

NIST Special Publication 800-53, issued by the National Institute of Standards and Technology, serves as the fundamental security control catalog for federal information systems. Two control groups are of paramount importance to our research:

- SI-2 (Flaw Remediation): This control mandates proactive lifecycle management in addition to patch deployment. It requires organizations to identify and remediate software defects within a defined timeframe. In our study, we evaluated the effectiveness of SI-2 by monitoring the residual risk remaining in the system following an official minimal ISO installation and the execution of available updates (via apt update && apt full-upgrade -y) followed by a mandatory system reboot.
- SC-7 (Boundary Protection): This establishes requirements for logical isolation and service segmentation. It provides the theoretical basis for comparing bare-metal operating systems with containerized environments. While boundary protection in bare-metal systems often relies solely on external firewalls, containerization (specifically lxc in our research) implements SC-7 requirements through internal, kernel-level isolation [3].

1.2. NIST SP 800-190

Given that the research is built upon Proxmox and lxc technologies, the NIST SP 800-190 Application Container Security Guide is also a decisive factor [5]. According to this framework, containerization presents both advantages and disadvantages. A primary advantage is the natural reduction of the attack surface due to the minimization of user-space content; consequently, containers do not include unnecessary drivers and, in most cases, lack redundant background processes. However, the shared kernel vulnerability poses a significant risk. If an attacker manages to achieve a "container escape", the entire host system – including all other containers – is jeopardized. The vulnerability dynamics examined in this paper specifically analyze how the rapid redeployability (immutability) of containers serves to offset these specific risks.

1.3. CIS Benchmarks

While NIST defines high-level security controls, the Center for Internet Security (CIS) prescribes concrete, technical-level configuration parameters. In applying the CIS Ubuntu Linux 24.04 LTS STIG Benchmark [1], as well as similar benchmarks for Debian Linux and Microsoft Windows 11, we examined the systems across more than 200 audit points aimed at hardening configuration settings. Throughout the

research, the CIS Compliance Score served as the primary comparative metric between the various OS variants.

1.4. CVSS

Detected CVEs (Common Vulnerabilities and Exposures) were evaluated not merely based on quantitative factors but were weighted according to the metrics of the Common Vulnerability Scoring System (CVSS) [2]. The CVSS v3.1 and v4.0 frameworks – which remain the most widely accepted standards despite their subjective elements – categorize the severity of vulnerabilities through a process based on the measurement of their specific characteristics. Consequently, this system is suitable for classifying flaws ranging from critical vulnerabilities to low-risk security issues requiring only local access.

2. Methodology and Data Collection

The methodological structure of this research is based on a quantitative comparative analysis conducted in two successive phases. The first phase involved a retrospective analysis within a controlled university network environment, which was subsequently expanded in the second phase to encompass real-world market environments: the infrastructures of commercial enterprises employing more than 50 people. This dual approach enabled the validation of theoretical vulnerability models under actual operational conditions.

The core of the empirical investigation focused on analyzing the correlation between system age, software 'freshness,' and detected critical vulnerabilities (CVEs). Our hypothesis posits that the default configurations of traditional operating systems contain redundant services and modules that unnecessarily increase the attack surface, potentially compounding risk over time (technical debt) compared to minimalist containerized environments.

For technical validation, the Wazuh open-source SIEM (Security Information and Event Management) and XDR (Extended Detection and Response) platform was utilized [6]. This tool was selected for its integrated capability to manage security events, configuration audits, and vulnerability detection.

The experimental setup was built on two primary pillars across the evaluated operating systems:

- Reference Group: Standard monolithic operating system installations (including Ubuntu, Debian, and Microsoft Windows 11 Pro) maintained within Virtual Machine (VM) environments using default, "out-of-the-box" settings. This represents average system administration practice, where rapid deployment takes precedence over fine-tuning.

- Test Group: Functionally equivalent Linux Container (LXC) instances (with the exception of Windows 11 Pro) running within a Proxmox VE environ-

ment, providing the same network services but benefiting from the isolation advantages inherent to the containerization layer.

2.1. Experimental Design and Architectural Scope

To ensure methodological clarity, the experimental configurations were subjected to standardized evaluation criteria. The security telemetry was gathered using Wazuh agents operating under their out-of-the-box, default scanning schedule configurations to reflect the original, non-customized deployment scenarios. The empirical observation was conducted over a 3-day window, during which the systems were evaluated in their initial deployment state, followed by the execution of a complete update cycle utilizing all available stable packages from official repositories.

The structural comparison involves a deliberate architectural boundary regarding Microsoft Windows 11 Pro. In this study, Windows 11 is evaluated exclusively as a Virtual Machine (VM) environment, without an equivalent Windows-based container instance. This scoping is mandated by the underlying technical infrastructure: the Proxmox VE hypervisor utilizes Linux-native kernel features (namespaces and control groups) to implement LXC (Linux Containers). Because Windows containerization relies on a fundamentally different container model and kernel abstraction layer requiring a Microsoft-native ecosystem, its inclusion fell outside the primary scope of this comparative framework. Therefore, the inclusion and identical evaluation of Windows 11 Pro (in terms of configuration compliance and vulnerability metrics) aims to provide a heterogeneous baseline, contextualizing the volume and severity of security flaws found within the Linux ecosystem against a widely deployed alternative enterprise operating system.

2.2. Assessment Parameters and Protocols

During the investigation, both the network and the local scanning protocols were applied, with particular focus on the following critical metrics:

- Attack Surface (Exposure Surface): We examined not only the number of open ports but also the versions and necessity of the underlying services. For instance, we analyzed Level 1 (highly recommended basic security hardening) and Level 2 (defense-in-depth) recommendations within the CIS Benchmarks, ensuring that unnecessary and insecure protocols (e.g., telnetd) which pose immediate security risks were strictly audited.
- Audit of Active Background Processes: We identified services that are not required for the specific system role but remain part of the default installation (e.g., avahi-daemon, or Bluetooth server processes in a server environment).
- CVE Detection: Using the Wazuh vulnerability database, we maintained a real-time inventory of known defects in installed packages, with special attention to critical-rated vulnerabilities (CVSS $>$ 9.0), such as those that affect the glibc or OpenSSL libraries [4].

- Hardening Drift: The most significant part of the measurement involved a comparison based on the CIS Ubuntu Linux 24.04 LTS STIG Benchmark, which quantifies the extent to which a system's configuration deviates from the optimal security baseline.

Data was recorded following the initial installation and the completion of the first update cycle. The systems involved in the study were:

- Proxmox VE 9.1
- Windows 11 Pro vm
- Ubuntu 24.04.3 vm
- Debian 12 vm
- Debian 13 vm
- Ubuntu 24.04.3 ct
- Debian 12 ct
- Debian 13 ct

3. Results

The data collected during the study suggest patterns aligned with a gradual vulnerability accumulation process over time, often described in the literature as security drift. As system uptime increases, configurations tend to diverge from their initial secure baseline, while the number of unpatched software vulnerabilities correspondingly increases.

Measurements focusing on configuration compliance (CIS Benchmark results) show that an out-of-the-box installation satisfies only a subset of the recommended industry security controls. Table 1 summarizes the CIS compliance indices for the evaluated environments immediately after system deployment and following the installation of available updates.

Based on the CIS Benchmark assessment results (Table 1), Ubuntu 24.04 achieved a compliance score of 50% in the virtual machine configuration and 51% in the lxc container environment, indicating only marginal improvement in the default hardened state. Debian-based systems demonstrated lower baseline compliance, ranging from 37% (VM) to 41% (container), while Proxmox VE itself achieved 40

In contrast, Windows 11 Pro exhibited a significantly lower compliance score of 24%, with 358 failed controls compared to 115 passed controls, highlighting a substantial deviation from CIS-recommended configurations in its default state. This discrepancy is largely attributable to the fact that a default Windows 11 installation includes numerous pre-installed consumer-oriented applications (bloatware) and active telemetry services, which inherently creates a broader attack surface

Table 1. CIS Benchmark compliance results across evaluated systems (vm vs lxc environments).

OS / Environment	Passed	Failed	Not appl.	Compliance score
Ubuntu 24.04.3 vm	119	118	42	50%
Ubuntu 24.04.3 ct	120	112	47	51%
Debian 12 vm	66	111	30	37%
Debian 12 ct	70	103	34	40%
Debian 13 vm	66	111	30	37%
Debian 13 ct	72	101	34	41%
PVE (Proxmox)	75	110	22	40%
Windows 11 Pro vm	115	358	9	24%

and lower baseline compliance compared to the 'minimal' installation profile used for the Linux distributions in this study. A consistent pattern can be observed across all tested Linux distributions: containerized (lxc) environments achieved slightly higher compliance scores than their virtual machine counterparts. For example, Debian 13 improved from 37% (VM) to 41% (container), while Debian 12 increased from 37% to 40%. Ubuntu 24.04 showed a smaller difference, increasing from 50% to 51%. Overall, the observed compliance differences between VM and container environments range between 1–4 percentage points, indicating that while containerization alone does not ensure high compliance, it provides a consistently improved baseline.

Table 2. A comparative overview of the vulnerability assessment results.

OS / Environment	Critical	High	Medium	Low
Ubuntu 24.04.3 vm	3	145	337	43
Ubuntu 24.04.3 ct	31	159	193	45
Debian 12 vm	0	192	789	6
Debian 12 ct	0	0	6	0
Debian 13 vm	0	14	58	0
Debian 13 ct	0	3	5	0
PVE (Proxmox)	0	0	8	19
Windows 11 Pro vm	2	212	76	0

During the analysis of software vulnerabilities, defects within installed packages were identified using the Wazuh SIEM platform. A notable observation is that containerized environments demonstrated improved outcomes following the update cycle. While certain container instances initially exhibited a higher num-

ber of critical vulnerabilities, targeted updates significantly reduced these values. In particular, Debian-based container instances reached a state where no critical vulnerabilities were detected by the applied scanning methodology.

The findings suggest that, under the tested conditions, containerized environments may offer certain security advantages over traditional monolithic system deployments. Based on the analysis, three primary contributing factors can be identified:

- Architectural Minimalism (Abstraction Level): container images typically include only the components required for a given function, reducing the presence of unnecessary software packages and services. In contrast, general-purpose operating systems often include a broader set of default components, which may increase the potential attack surface.
- Logical Isolation: namespace-based isolation and control groups (cgroups) provide separation between workloads. These mechanisms can limit the impact of a potential compromise and may reduce the likelihood of lateral movement compared to less segmented environments.
- Immutability and Update Strategy: container-based workflows support the concept of immutable infrastructure, where systems are redeployed from updated images rather than modified in place. This approach can contribute to maintaining alignment with security baselines and supports practices aligned with NIST SP 800-53 SI-2 (Flaw Remediation).
- Overall, the results indicate that the security characteristics of a system are significantly influenced by architectural design choices, particularly in relation to component minimalism, isolation mechanisms, and update strategies.

4. Conclusion

The findings of this study suggest that vulnerability management should not be regarded solely as an operational maintenance activity, but rather as a function strongly influenced by architectural design decisions. The results indicate that configuration drift and vulnerability accumulation are observable characteristics of traditionally managed systems, which may be mitigated through alternative deployment models. Within the scope of the examined environments, container-based approaches demonstrated favorable security characteristics compared to monolithic system deployments. However, these advantages are context-dependent and rely on appropriate configuration and operational practices.

Based on the observations, the following strategic considerations can be formulated:

- Modernization Strategy: containerized deployment models may provide a viable alternative to traditional in-place upgrades of legacy systems. Under the tested conditions, functionally equivalent lxc containers exhibited fewer

critical vulnerabilities after the update cycle than comparable virtual machine instances. This difference appears to be associated with reduced component complexity and a smaller effective attack surface.

- Hardening by Design: the use of pre-configured and security-hardened templates can reduce the initial compliance gap observed in default system installations. Embedding CIS-aligned configurations into deployment artifacts may support more consistent adherence to security baselines from the outset.
- Immutability and Auditability: adopting immutable infrastructure principles – where systems are redeployed rather than modified – can improve traceability and help maintain alignment with established security controls, including practices consistent with NIST SP 800-53 SI-2. This approach may reduce the long-term effects of configuration drift observed in traditionally managed environments.

At the same time, it is important to emphasize that the security benefits of containerization are not inherent but depend on correct implementation. Misconfigured containers (e.g., excessive privileges or inadequate isolation settings) may negate these advantages and introduce additional risks.

Finally, this study has several limitations. The results are based on a specific technological stack (Proxmox VE and lxc), a limited set of operating systems, and a defined observation window. Consequently, the findings should be interpreted within this context; given the 3-day observation window, these results reflect immediate post-deployment dynamics rather than long-term risk progression, and further research is required to evaluate their generalizability.

Future work should extend the analysis to additional containerization platforms (e.g., Docker, Kubernetes) and include longer observation periods to better characterize vulnerability lifecycle dynamics.

References

[1] Center for Internet Security: *CIS Benchmarks: Ubuntu Linux 24.04 LTS STIG Benchmark*, URL: https://www.cisecurity.org/cis-benchmarks (visited on 01/29/2026).

[2] FIRST.org, Inc.: *Common Vulnerability Scoring System (CVSS)*, URL: https://www.first.org/cvss/ (visited on 01/29/2026).

[3] O. Jarkas et al.: *A Container Security Survey: Exploits, Attacks, and Defenses*, ACM Computing Surveys 57.7 (2024), DOI: 10.1145/3715001.

[4] Jumiaty, B. Soewito: *SIEM and Threat Intelligence: Protecting Applications with Wazuh and TheHive*, International Journal of Advanced Computer Science and Applications 15.9 (2024).

[5] National Institute of Standards and Technology: *Special Publication 800-190: Application Container Security Guide*, 2017, URL: https://csrc.nist.gov/pubs/sp/800/190/final (visited on 01/29/2026).

[6] S. Stanković, S. Gajin, R. Petrović: *A Review of Wazuh Tool Capabilities for Detecting Attacks Based on Log Analysis*, in: Proceedings of the IX International Conference, 2024.