OS-Level Vulnerabilities in Virtualized Environments: Examination, Mitigation, and Future Directions

Ishaan Sharma
MS in Computer Science
University of Southern California, Los Angeles, California
ishaansh@usc.edu

***Abstract*—Adopting virtualization technologies, such as Virtual Machines (VMs) and containers, has revolutionized modern computing. With this technology, organizations have attained greater scalability and flexibility and better optimized resources. Virtualization allows multiple isolated systems to share the same physical hardware, thus reducing operational costs and enhancing deployment efficiency. On the other hand, this multi-level architecture provides some OS-level-specific vulnerabilities critical to the same benefit of virtualization it is introduced to deliver. This includes isolation failure, hypervisor exploits, container escapes, side-channel attacks, and more. All these raise significant concerns about a virtualized environment's confidentiality, integrity, and availability.**

This research seeks to explore the nature and scope of OS-level vulnerabilities within virtualized environments, identifying the specific attack vectors that make them susceptible to compromise. The study will critically analyze existing mitigation strategies, including isolation techniques, monitoring tools, and security hardening practices, to assess their effectiveness and limitations. Furthermore, the paper aims to propose innovative solutions and improvements, drawing on emerging technologies such as artificial intelligence-driven threat detection, hardware-assisted security mechanisms, and enhanced isolation protocols. By addressing these vulnerabilities and exploring novel directions, this research aspires to contribute valuable insights for strengthening the security of virtualized environments, ensuring their reliability in a landscape increasingly dependent on virtualization technologies.

***Index Terms*—Virtual Machines (VMs), Hypervisor Security, Virtualization Vulnerabilities, Side-Channel Attacks, OS-Level Vulnerabilities, Hardware-Assisted Security, AI in Cybersecurity, Machine Learning for Threat Detection, Blockchain Auditing, Intrusion Detection Systems (IDS), Container Security, Policy Management, Zero-Trust Architecture, Secure Configurations, Cloud Security.**

I. Introduction

In the past few years, virtualization environments have become essential in various domains for testing and development, server consolidation, running legacy applications, disaster recovery, digital forensics, and cloud computing. Rising usage of virtualization software is primarily driven by its cost effectiveness, flexibility, and general efficiency. Virtualization technology begins as far back as the 1960s when IBM came with the idea of virtual machine technology on mainframes, allowing for resource optimization and the ability to run multiple operating systems concurrently. Companies such as VMware, Citrix, and Microsoft pushed x86 virtualization by the early 2000s, firmly embedding it in modern cloud computing and IT infrastructure as today.

TABLE I
Abbreviations

Abbreviation	Full Form
VM	Virtual Machine
TEE	Trusted Execution Environment
IDS	Intrusion Detection System
SEV	Secure Encrypted Virtualization
NPT	Nested Page Tables
POLP	Principle of Least Privilege
OS	Operating System
RDP	Remote Desktop Protocol
SLAT	Second Level Address Translation
CVE	Common Vulnerabilities and Exposures

Increased virtualization does have its side effects of making it an lucrative target for sophisticated security threats. Virtualized environments commonly hold critical infrastructure or sensitive data making them attractive to attackers. Vulnerabilities like VM Escape, Hyper-jacking, Downfall (CVE-2022-0847), Dirty Pipe (CVE-2022-0847), ZombieLoad (CVE-2018-12130), and Spectre/Meltdown may cause a breach in the isolation between virtual machines or between a virtual machine and the host, thereby causing data breaches, unauthorized access, and system disruption. These threats indicate the urgency to have strong security measures protecting virtualized systems.

This paper identifies a few of these vulnerabilities, explores the existing mitigation strategies, and mentions future directions in securing virtualization environments.

II. Related Work

A. A Survey on Security Challenges of Virtualization Technology in Cloud Computing

This paper emphasizes on the security challenges that virtualization technology introduces in cloud computing environments. It underlines the vulnerabilities that arise both from guest operating systems and hypervisors, emphasizing the role of VM escape as a major security threat. The paper explores how, despite virtualization's advantages, the additional abstraction layer complicates security controls, increasing the system's vulnerability. Key measures for mitigating these risks include regular updates, patch management, and rigorous isolation policies. Such measures are essential for maintaining security in complex cloud-based systems where multiple VMs coexist[25].

B. Understanding VM Escape: A Threat to Virtualized Environments

This paper dives deeply into VM Escape attacks, explaining how adversaries exploit vulnerabilities to break out from a Virtual Environment, potentially compromising the entire host and other co-hosted virtual machines. The focus is on the techniques used by adversaries, such as remote code execution and also how they leverage shared resource vulnerabilities that disrupt the security of the virtualization software. It also shares with us the consequences of VM Escape are significant, leading to potential data breaches, denial of service attacks, and compromise of sensitive information. This paper underlines the necessity for a robust system for hypervisor security practices and constant vigilance to mitigate the risk associated with VM escape.

C. Unveiling the Landscape of Operating System

This paper provides insights into common operating system vulnerabilities in virtualized environments, particularly focusing on those affecting guest operating systems. Denial of Service attacks are highlighted as a pervasive threat, exploiting resource distribution flaws to degrade functionality. The paper emphasizes vulnerabilities arising from resource isolation issues, insufficient patch management, and poor authentication. The implications for security are severe, necessitating a comprehensive and proactive approach to security management, including regular updates, network monitoring, and enhanced access controls to prevent unauthorized access and exploitation.

D. SEVerity Code Injection Attacks against Encrypted Virtual Machines

This paper exposes critical vulnerabilities within AMD's Secure Encrypted Virtualization (SEV) infrastructure, chiefly pointing out flaws in the SEV Encrypted State (SEV-ES) functionality. The primary concern is SEV-EV's inability safeguard memory integrity of VMs, exposing systems to potential code injections by adversaries with control over the hypervisor. This gap allows adversaries to delegate malicious code via I/O channels, going around the encryption safeguards by embedding payloads in network communications routinely processed by the VM. SEV's weakness is amplified by by inadequate separation within shared resources, permitting cross-domain interference through shared memory utilized in I/O processes. This oversight enables unauthorized data interaction, particularly through shared page tables, leading to severe security breaches. The paper's experimental results demonstrate a consistent 100% success rate in executing the SEVerity attack, manifesting the urgent need for SEV design enhancements. The capacity for malicious entities to execute arbitrary code within VMs signifies a considerable threat, potentially resulting in data exfiltration or manipulation under the guise of legitimate operations. Consequently, the study underscores the necessity for refined memory integrity protocols and fortified access control mechanisms to secure encrypted virtualized machines comprehensively.

III. Examination: OS-Level Vulnerabilities

The Examination of OS-Level vulnerabilities in virtualized environments reveals various categories of risks that can compromise the security of systems. These vulnerabilities primarily arise from issues related to isolation failures, hypervisor exploits, and side-channel attacks.

The OS-Level Vulnerabilities can be broadly classified into three categories
1. Isolation Failures
2. Hypervisor Exploits
3. Side Channel Attacks

*A. **Isolation Failures***

*1) **Understanding Isolation in Operating System**:* Isolation is the cornerstone of all the modern operating system security. Isolation ensures that all the processes, applications, and virtual environments operate independently, preventing unauthorized access and interference. The isolation mechanisms include:

- Process Isolation: Keeping processes in separate memory spaces.
- User and Kernel Modes: Segregating user applications from core kernel system functions.
- Virtual Machines: Uses hypervisors to create isolated environments.
- Containers: Provides application-level isolation using shared OS kernels.

Failures in these isolation mechanisms is what is called Isolation Failures and these failures lead to severe security breaches, allowing attackers to access sensitive data, Escalate Privileges, Compromise Systems.

*2) **Isolation Failure Attacks**:*

- **VM Escape**
  VM Escapes are security vulnerabilities that allow an adversary to break out of a virtual environment and gain access to the host system or other virtual machines. This escape of adversary from the virtual environment to host pose some significant risks, especially in environments that rely heavily on virtualization for security isolation.
  Some notable VM Escape Vulnerabilities:
  1) **Microsoft Hyper-V Remote Code Execution Vulnerability (CVE-2021-28476):** Because of this flaw, a guest virtual machine can make the Hyper-V host's kernel read from any random, possibly

invalid address. The guest virtual machine would not receive the contents of the address read. The Hyper-V host would often experience a denial of service (bug check) as a result of reading an unmapped address. The security of the Hyper-V host was jeopardized if additional, hardware device-specific side effects are triggered by reading from a memory-mapped device register corresponding to a hardware device connected to the Hyper-V host.

2) **VMware's Authentication Bypass Vulnerability (CVE-2022-22972):** VMware Workspace ONE Access, Identity Manager and VRealize Automation contained an authentication bypass vulnerability affecting local domain users. This attack was carried out by using a malicious actor that had access to the UI through network, can gain administrative access without ever needing to authenticate.
3) **KVM Privilege Escalation (CVE-2019-3016):** This was a vulnerability in the kernel-based Virtual Machine (KVM) for Linux, This vulnerability allowed a guest VM to read memory from the host kernel. This exploit took advantage of flaw in the memory management system of KVM. Due to this exploit the isolation between the guest and the host machine was broke that led to data leakage.
4) **Azure Cloud Escape - CVE-2021-28482):** A flaw in the Azure Sphere operating system's handling of system calls allowed code execution on the Azure Sphere Security Monitor. It's impact was it broke the isolation between the user application and the security monitor which led to unauthorized access to secure components of the device. Which in turn potentially compromised the whole device.

B. *Hypervisor Exploits*

*1) **Understanding what is Hypervisor**:* A Hypervisor, also known as Virtual Machine Monitor (VMM), is a software that create a manages VMs by dividing the hardware resources. This software enables multiple operating systems to share a single hardware host, providing isolation between VMs. There are two type of main hypervisors:

- Type 1 Hypervisors (Bare-Metal): These run directly form host's hardware. E.g. VMware ESXi, Xen.
- Type 2 Hypervisors (Hosted): These run on top of a host's OS. For e.g. VMware Workstation, Oracle Virtualbox.

Exploiting vulnerabilities in an Hypervisor can lead to severe security breaches, as hypervisors are known to have high privileges and total control over multiple VMs. Exploiting these vulnerabilities can result in Denial of Service, VM Escape, Data Leakage.

*2) **Hypervisor Attacks**:* Hypervisor exploits occur due to the flaws in the codebase of the hypervisor, issues in the device emulation used by hypervisors, CPU vulnerabilities that affect the security of hypervisor, configuration mismatch can expose interfaces to adversaries and last but not least Exploitation of shared memory or hardware resources.

Some of recent hypervisor attacks

1) **BlueKeep RDP Vulnerability (CVE-2019-0708):** BlueKeep is a critical vulnerability in Remote Desktop Services, while this one was not particularly an hypervisor exploit, but some environments where RDP was being used to manage the hypervisors in the OS, it lead to compromise the hypervisor security. This was caused by the per-authentication remote code execution vulnerability, which in turn allowed the attacker to execute arbitrary code in the target system.
2) **Hyper-V Denial of Service (CVE-2021-28445:)** This is a vulnerability in Hyper-V network virtualization that allows a guest VM to cause the host to crash. This vulnerability is caused by the improper handling of the network packets, which in turn leads to system crash initiated from the guest VM denying the access to OS, hence Denial of Service attack.
3) **ESXi OpenSLP Heap-Overflow Vulnerability (CVE-2020-3992):** This was a heap-overflow vulnerability in the OpenSLP service of VMware ESXi, which allowed an adversary to have access to port number 427, which allowed them to execute arbitrary code remotely onto the hypervisor. This vulnerability was caused by improper handling of network packets.
4) **AMD SEV Vulnerabilities (CVE-2019-9836):** This vulnerability in AMD's Secure Encrypted Virtualization was caused by there weak implementation of their SEV framework which allowed attacker to bypass the memory encryption protection layers, The attacker was able to do so by bypassing AMD Secure Encrypted Virtualization with Encrypted State (SEV-ES) protections by targeting the lack of memory integrity protection and Second Level Address Translation (SLAT). Which in turn allowed attackers to remap guest physical memory frames (GFNs) to system frames (SFNs) of their choice.

TABLE II
SUMMARY OF VIRTUALIZATION VULNERABILITIES

Vulnerability	Description
VENOM	Buffer overflow in virtual FDC
Hyper-V RCE	CredSSP protocol vulnerability
Xen Null Pointer	Dereference in PCI passthrough
KVM Privilege Escalation	Flaw in vhost-net module
ESXi OpenSLP	Heap-overflow in OpenSLP service
L1TF	Speculative execution side-channel
AMD SEV Flaws	Weakness in memory encryption
Hyper-V DoS	Network virtualization vulnerability
QEMU USB Vulnerability	Out-of-bounds access in USB redirector
BlueKeep	RDP vulnerability leading to code execution

C. *Side-Channel Attacks*

Side-channel attacks are sophisticated techniques that exploit indirect information leaks from a system, rather than targeting the system's code directly. These attacks leverage physical phenomena such as power consumption, electromagnetic emissions, timing information, and processor cache behavior to deduce sensitive data. In the context of operating system (OS)-level vulnerabilities, side-channel attacks can compromise highly secure environments by exploiting shared resources like CPU caches, which is particularly prevalent in virtualized environments where multiple systems may share underlying hardware.[4]

*1) **Working of Side-Channel Attacks**:*

- Information Extraction: Side-channel attacks do not go breaching software defenses; instead, they work by doing reconnaissance first, they gather information by measuring how a system's operation will affect its hardware environment. For example, variation in the execution time can reveal the logical paths taken by cryptographic operations, which can lead to the recovery of secret keys.[5]
- Exploiting Shared Resources: In multi-tenant environments such as cloud services, side-channel attacks often exploit shared hardware resources. For example, an attacker can analyze the CPU cache usage patterns of a victim virtual machine (VM) to deduce the types of computations it is performing[6]
- Undetected Exploitation: Because these attacks do not alter fundamental system operations, they often go undetected by conventional security measures. This makes them particularly insidious in environments where VMs share hardware, as an attacker VM can covertly collect data from a victim VM residing on the same host[7]

*2) **Common types of Side-Channel Attacks**:*

- Cache Timing Attacks
- Power Analysis
- Electromagnetic Analysis
- Acoustic Cryptanalysis
- Rowhammer

3) Side-Channel Vulnerabilities:

- **Rowhammer** This attack rapidly toggles rows in DRAM that causes bit flips in adjacent rows, potentially altering data in unauthorized areas of memory, This type of attack can cause privilege escalation, corruption of critical OS data, and cross-VM attacks in shared environments.
- **ZombieLoad (CVE-2018-12130)** This attack exploits micro architectural data sampling (MDS) in Intel CPUs to access recently used data in the CPUs buffer. This attack leaks sensitive information like passwords, encryption keys, or browsing history.
- **Foreshadow (L1 Terminal Fault)** Exploits vulnerabilities in Intel SGX (Software Guard Extensions) to read data from CPU's L1 cache. Exploiting this vulnerabilities gives access to sensitive data intended to protect SGX.
- **Spectre (CVE-2017-5753 and CVE-2017-5715)** Exploits speculative execution in modern CPUs to trick a program into accessing a memory that would be normally out of bounds. This attack allows attackers to infer memory contents across process boundaries, such as sensitive user data.

IV. Analysis of Existing Mitigation Strategies

A. Sandbox Environments

1) Overview: Sandbox environments provide a contained and secure area for executing untrusted or potentially malicious code, avoiding system-wide repercussions. They create a controlled virtualization layer that isolates applications from the underlying operating system, preventing unwanted system interactions and contamination.[8]

2) Effectiveness and Limitation:
Effectiveness: Sandboxing is highly effective in scenarios where it's crucial to protect the host environment from potentially harmful software[8]. Successful deployments by Adobe and Google have demonstrated that reducing privilege levels within sandboxed processes significantly lowers the risk of exploits.[9]

Limitations: The overhead of virtualization can lead to performance degradation, making sandboxing less practical for resource-intensive applications. Sandboxing reliance alone is insufficient as advanced threats continue to develop methods to detect and evade sandbox environments.[8]

3) Real-World Cases: In 2010, Adobe's implementation of sandboxing in Acrobat led to a reduction in vulnerabilities, correlating directly with a decrease in exploit attempts[8]. However, advanced threats presented at the Black Hat Europe event demonstrated that even sandboxed environments could be manipulated, suggesting the need for continuous improvement[8].

B. Namespace Use in Containers

1) Overview: Namespaces in Linux are a crucial component of container security, providing process isolation by restricting access to resources and separating environments within a host system.[10]

2) Effectiveness and Limitations:
Effectiveness: Namespaces effectively compartmentalize processes, limiting their abilities to interfere with other processes on the same host[10]. This modular separation is particularly beneficial for maintaining efficiency and security in containerized applications[10].

Limitations: Misconfiguration or inadequate use of namespaces can lead to weakened isolation, providing opportunities for attackers to exploit shared resources[10]. Ensuring configurations are properly managed is critical[10].

3) Real-World Cases: The Docker containerization platform demonstrates the effective use of namespaces by isolating file systems and networks within containers, minimizing the risk of system-wide exploits[10]. However, cases of misconfiguration where containers are inadvertently given root access have underscored the necessity for rigorous namespace handling and security measures[10].

C. Timely Patching

1) Overview: Timely Patching is fundamental to maintaining system security. It involves the prompt application of updates and patches to eliminate vulnerabilities[11]

2) Effectiveness and Limitations:
Effectiveness: Regular patching is a cornerstone of cybersecurity, crucial for mitigating known vulnerabilities and safeguarding against exploits[11]. Statistics indicate that over 90% of breaches involve unpatched vulnerabilities[12].

Limitations: Despite its importance, timely patching faces challenges such as resource constraints and patch compatibility issues, which can delay deployment and leave systems exposed to threats[13].

3) Real-World Cases: Organizations that implemented automated patch management, such as Acronis, have shown a significant decrease in patching time and an improved security posture[11]. Conversely, delayed patch deployments have led to successful breaches, highlighting the potential pitfalls of inadequate patch management.

D. Least Privilege Policies

1) Overview: The principle of least privilege (POLP) grants users and applications only the access necessary to perform their functions, reducing the potential impact of attacks[14].

2) Effectiveness and Limitations:
Effectiveness: Implementing POLP enhances security by minimizing the attack surface, preventing unauthorized access, and containing malware propagation[14].

Limitations: The main challenge lies in the balance between security and operational efficiency, as over-restrictive policies can impede user productivity and lead to privilege creep if not managed diligently[15].

3) Real-World Cases: The AMCA breach exemplifies the failure of least privilege when third-party access is not properly controlled, leading to significant data loss[15]. Conversely, organizations employing POLP effectively manage risk by restricting unnecessary administrative access, preventing widespread breaches[14].

E. Intrusion Detection System (IDS)

1) Overview: Intrusion Detection Systems (IDS) monitor network or system activities for malicious actions or policy violations, providing alerts for administrative intervention[16].

2) Effectiveness and Limitations:
Effectiveness: IDS offers enhanced visibility across networks, helping in the detection of unauthorized access and potential threats before any severe damage occurs.

Limitations: IDS can produce false positives, generating alerts for benign activities, requiring further analysis and potentially leading to reduced effectiveness if not managed properly[17].

3) Real-World Cases: Network-based IDS have successfully detected protocol-based exploitation attacks in organizations by analyzing network traffic in real-time[17]. However, their inability to process encrypted packets poses a significant limitation, allowing certain attacks to go undetected[17].

F. AWS Inspector

1) Overview: AWS Inspector is a vulnerability management tool that continuously scans AWS environments for potential security issues, providing insight into exposure and protection status[18].

2) Effectiveness and Limitations:
Effectiveness: AWS Inspector is highly rated for its ability to discover and address software vulnerabilities, ensuring compliance with security benchmarks and standards[19].

Limitations: While effective within AWS ecosystems, its utility in hybrid cloud environments is limited, necessitating integration with other third-party tools for comprehensive security[19].

3) Real-World Cases: In a study, AWS Inspector detected 95% of known vulnerabilities within test environments, highlighting its efficacy in securing AWS cloud infrastructures[19]. Its integration into DevSecOps pipelines has further demonstrated its value in enhancing security posture through continuous assessment[19].

G. AppArmor

1) Overview: AppArmor is a Linux security framework that uses profiles to limit the capabilities of programs, reducing risk exposure by defining specific access permissions for applications.

2) Effectiveness and Limitations:
Effectiveness: AppArmor provides an easy-to-manage framework for application confinement, offering protection by restricting program operations more intuitively than SELinux.

Limitations: Misconfigurations within AppArmor profiles can lead to security vulnerabilities, as demonstrated by occasional policy loopholes in mount operations.

3) Real-World Cases: Instances where AppArmor successfully restricted application behavior—such as in the Ubuntu environment—show the importance of a structured security framework in preventing unauthorized access[20]. Misconfigurations, however, have occasionally led to increased attack surfaces[21].

V. Proposed Improvements and Future Directions

Virtualization security is an ongoing challenge that necessitates a proactive approach to mitigating risk and enhancing security. As such, several improvements to existing strategies can be proposed, along with future directions that embrace emerging technologies and trends. This section outlines advanced isolation techniques, the application of artificial intelligence (AI) and machine learning (ML) for threat detection, policy recommendations, and novel security approaches.

1) Advanced Isolation Techniques: Strong hardware-assisted virtualization technologies, like Intel VT-x, give a robust foundation for security enhancement in virtualized environments. Such technologies allow better isolation of VMs by exploiting hardware capabilities to enforce boundaries between them. Implementation of Nested Page Tables (NPT) can also ensure that each VM operates under a strictly defined memory space, thereby reducing the risk of unauthorized memory access or data leakage[20].

Moreover, the development and integration of Trusted Execution Environments (TEEs) present an opportunity for improving isolation. TEEs can execute code in a secure environment isolated from the main operating system, effectively protecting sensitive operations from potential threats within the hypervisor[21]. By implementing these advanced techniques, organizations can strengthen their defenses against privilege escalation vulnerabilities that have been prevalent in conventional virtualization setups.

2) AI and ML for Threat Detection: The rise of AI and ML presents significant opportunities for augmenting security measures in virtualized environments. Using machine learning models for anomaly detection can identify unusual patterns in system behavior that may indicate a security breach. For instance, a supervised learning model can be trained on normal operational data to recognize deviations that deviate from established baselines[22].

Hypothetical scenarios illustrate the application of this approach: In a case where a VM exhibits sudden spikes in CPU usage, the model can flag this behavior as an anomaly, triggering automated responses such as resource isolation or alerting security personnel for further investigation. The integration of AI-driven threat intelligence can also facilitate real-time data analysis, enabling organizations to promptly respond to emerging threats before they lead to system compromises[23].

3) Policy Recommendations: Policy frameworks play a critical role in maintaining secure virtualization infrastructures. Guidelines for patch management should emphasize the importance of timely updates and vulnerability assessment, specifically focusing on hypervisors and guest operating systems[24]. Conducting audits regularly is very important in ensuring that their policies are followed and security configurations adhered to.

In addition to patch management, organizations should adopt secure configuration practices that involve hardening both the host and virtual machines. This includes disabling unnecessary services, utilizing strong access controls, and applying network segmentation to isolate critical resources from general network traffic. By implementing these guidelines, organizations can significantly reduce their attack surfaces and improve overall resilience against potential threats.

4) Security-by-Design Approaches: Incorporating security by design principles entails embedding security measures and best practices into the software development lifecycle from the outset, rather than as an afterthought. For virtualization technologies, this can involve rigorous testing of new features for security vulnerabilities before deployment and adopting a risk management approach throughout the development phases.

Having a consistent review process for security implications allows organizations to preemptively address vulnerabilities inherent in the virtualization architecture. Establishing security as a foundational element ensures that every layer, from hardware to software, contributes to the overall security posture, thus mitigating risks associated with virtual machine migrations and resource sharing.

5) Blockchain for Resource Auditing: The application of blockchain technology can provide new avenues for auditing and resource management within virtualized environments. By establishing a decentralized ledger for recording transactions and interactions, organizations can ensure transparency and immutability of data pertaining to resource allocation and access control.

Blockchain can also facilitate identity and access management, ensuring that only authorized entities can make modifications to the virtualized resources. For instance, implementing a smart contract system may govern VM lifecycle processes such as instantiation and termination, automatically enforcing security policies through programmable logic. This not only enhances security audits but also offers organizations a means to verify compliance with regulatory standards in real-time.

VI. Conclusion

Adopting these proposed improvements and future directions can significantly enhance virtualization security. By investing in advanced isolation techniques, leveraging AI and ML for proactive threat detection, establishing systematic policies, implementing security-by-design principles, and embracing blockchain technology, organizations can create a more robust security infrastructure. These strategies not only address immediate security concerns but also position organizations to effectively counter evolving threats in the digital landscape. As virtualization continues to expand and become integral to modern IT infrastructure, prioritizing these measures will be critical for safeguarding sensitive data and ensuring operational integrity.

Acknowledgment

I express my deep gratitude to Professor Clifford Neuman for providing invaluable guidance in selecting and being flexible with research and offering constructive feedback on crafting a survey paper, significantly enhancing the quality of this survey.

References

[1] Alameri, I., & Radchenko, G. (2017). Development of student information management system based on cloud computing platform. Journal of Applied Computer Science & Mathematics, 11, 9–14.

[2] Zhu, G., Yin, Y., Cai, R., & Li, K. (2017). Detecting virtualization-specific vulnerabilities in cloud computing environment. In IEEE International Conference on Cloud Computing, CLOUD 2017-June, 743–748.

[3] Sosinsky, B. (2011). Cloud computing bible. Wiley.

[4] Gavin Wright. (2021). What is a side-channel attack? TechTarget. https://www.techtarget.com/searchsecurity/definition/side-channel-attack

[5] Contributors to Wikimedia projects. (2024). Side-channel attack - Wikipedia. Wikimedia Foundation, Inc. https://en.wikipedia.org/wiki/Side-channel-attack

[6] Su, C., & Zeng, Q. (2021). Survey of CPU Cache-Based Side-Channel Attacks: Systematic Analysis, Security Models, and Countermeasures. Security and Communication Networks. https://doi.org/10.1155/2021/5559552

[7] Dark Reading Staff. (2023). Researchers Develop Cross-VM Side-Channel Attack. Dark Reading. https://www.darkreading.com/cyberattacks-data-breaches/researchers-develop-cross-vm-side-channel-attack

[8] What is Sandboxing? Understand Sandboxing in Cyber Security - OPSWAT. (2024). OPSWAT. https://www.opswat.com/blog/what-is-sandboxing

[9] Dark Reading Staff. (2023). The Pros And Cons Of Application Sandboxing. Dark Reading. https://www.darkreading.com/cyber-risk/the-pros-and-cons-of-application-sandboxing

[10] Rory McCune Senior Advocate - Security & Compliance. (2024). Container security fundamentals part 2: Isolation & namespaces — Datadog Security Labs. datadoghq.com. https://securitylabs.datadoghq.com/articles/container-security-fundamentals-part-2/

[11] Acronis. (2024). What is security patching? Best practices and importance. Acronis. https://www.acronis.com/en-eu/blog/posts/security-patching-complete-guide/

[12] Andrios Robert. (2023). 5 Real-world Examples of Effective Patch Management in System Administration. hoop.dev - AI-POWERED PRIVILEGED ACCESS TO DATABASES AND SERVERS. https://hoop.dev/blog/5-real-world-examples-of-effective-patch-management-in-system-administration/

[13] Lauren Ballejos. (2024). Risks of Delayed Patching: A Guide to Fix Slow Patching — NinjaOne. NinjaOne. https://www.ninjaone.com/blog/risks-of-delayed-patching/

[14] Team Copado. (2024). What are the Benefits of Principle of Least Privilege (POLP) for My Organization? Copado. https://www.copado.com/resources/blog/what-are-the-benefits-of-principle-of-least-privilege-polp-for-my-organization

[15] Least Privilege Access — The Least Privilege Policy Explained — Delinea. (2024). Delinea. https://delinea.com/what-is/least-privilege

[16] GeeksforGeeks. (2024). Intrusion Detection System (IDS) - GeeksforGeeks. GeeksforGeeks. https://www.geeksforgeeks.org/intrusion-detection-system-ids/

[17] Rapid7. (2024). The Pros & Cons of Intrusion Detection Systems — Rapid7 Blog. Rapid7 Blog. https://www.rapid7.com/blog/post/2017/01/11/the-pros-cons-of-intrusion-detection-systems/

[18] Configuration and vulnerability analysis in Amazon Inspector Classic - Amazon Inspector Classic. (2024). amazon.com. https://docs.aws.amazon.com/inspector/v1/userguide/security-vulnerability-analysis-management.html

[19] Servifyspheresolutions. (2024). AWS Inspector. Medium. https://medium.com/@servifyspheresolutions/aws-inspector-6e9fddb4fa86

[20] Jithin, R., & Chandran, P. (2014). Virtual machine isolation: A survey on the security of virtual machines. https://link.springer.com/chapter/10.1007/978-3-642-54525-2_8

[21] Shkoukani, M., & Murrar, S. (2024). SECURE AND ISOLATED COMPUTING IN VIRTUALIZATION AND CLOUD ENVIRONMENTS: A SYSTEMATIC REVIEW OF EMERGING TRENDS AND TECHNIQUES. Journal of Theoretical and Applied Information Technology.

[22] Sierra-Arriaga, F., Branco, R., & Lee, B. (2020). Security issues and challenges for virtualization technologies. ACM Computing Surveys (CSUR). https://dl.acm.org/doi/abs/10.1145/3382190

[23] Nemati, H. (2005). Secure system virtualization: End-to-end verification of memory isolation. arXiv Preprint arXiv:2005.02605. https://arxiv.org/abs/2005.02605

[24] Pearce, M., Zeadally, S., & Hunt, R. (2013). Virtualization: Issues, security threats, and solutions. ACM Computing Surveys (CSUR). https://dl.acm.org/doi/abs/10.1145/2431211.2431216

[25] M. Almutairy, N., & H. A. Al-Shqeerat, K. (2019). A Survey on Security Challenges of Virtualization Technology in Cloud Computing. International Journal of Computer Science and Information Technology. https://doi.org/10.5121/ijcsit.2019.11308