web of semantic

Journal of Interdisciplinary Science

Research Article Peer-review, Open Access

Architecting Cost-Effective CDOM to Cloud VM Transitions

Seraphine Korr [1]

Abstract

The transition from traditional SPARC-based Control Domains (CDOMs), a foundational virtualization mechanism in Oracle Solaris environments, to modern cloud-based virtual machines (VMs) represents a strategic shift for organizations seeking agility, scalability, and cost efficiency. CDOMs, while robust and deeply integrated with LDOM infrastructure, are increasingly viewed as legacy systems in today's hybrid and multi-cloud IT landscape. The rising operational expenses of maintaining aging SPARC hardware, limitations in elasticity, and difficulties integrating with modern DevOps pipelines are motivating enterprises to explore cloud migration strategies that maintain performance while significantly reducing total cost of ownership (TCO).

This review article examines architectural, financial, and operational considerations central to the successful migration from CDOM-based virtualization stacks to cloud-native VM environments. It begins with an analysis of CDOM design, highlighting key dependencies on physical I/O domains and the SPARC hypervisor. It then evaluates the suitability of target platforms such as Oracle Cloud Infrastructure (OCI), AWS EC2 (with Solaris-compatible images), and custom Azure environments, with attention to hypervisor compatibility, image formats, and network abstraction layers.

A core focus is placed on cost optimization, including strategies like reserved instances, licensing portability (BYOL), deduplicated storage, and network egress planning. The article further explores automation tooling for image conversion, Infrastructure as Code (IaC) deployment with Ansible and Terraform, and post-migration monitoring using Prometheus or OCI Monitoring. Security and compliance are addressed through IAM mapping, log auditing, and standards alignment with frameworks such as HIPAA and PCI-DSS.

Through real-world case studies in the manufacturing, finance, and government sectors, this review demonstrates phased migration strategies that minimize downtime, enforce regulatory controls, and reduce operational friction. Challenges such as unsupported Solaris versions, driver compatibility, and skills gaps in cloud-native architectures are discussed, alongside mitigation strategies that include training, emulation, and containerization.

Looking ahead, the article outlines future opportunities including AI-driven cost tuning, Solaris application containerization, and potential serverless replacements for stateless CDOM services. Ultimately, the review provides a roadmap for architecting cost-effective, resilient, and secure cloud VM transitions from legacy Solaris CDOM environments, enabling enterprises to modernize without compromising reliability or governance.

Keywords: CDOM; LDOM; Cloud VM; Oracle Solaris; SPARC Virtualization; Cloud Migration Strategy; Infrastructure as Code; Terraform; Ansible; Virtualization Architecture; Hybrid Cloud; Cost Optimization; Cloud Monitoring; Licensing Portability; VM Image Conversion; ZFS; Network Redesign; Compliance; Serverless Migration; AI-driven Cloud Operations.

[1] *Cloudspire Institute*

Web of Semantics: Journal of Interdisciplinary Science (2023)
https://wom.semanticjournals.org

1. Introduction

1.1 Background and Rationale

Control Domains (CDOMs) have long served as the administrative and operational backbone for Oracle SPARC servers running Logical Domains (LDOMs). As the hypervisor host, the CDOM manages system-wide resources including CPUs, memory, virtual disks, and networking for guest LDOMs. While this architecture has proven reliable for mission-critical workloads, particularly in financial services, manufacturing, and government environments, its reliance on specialized hardware and legacy virtualization limits scalability, integration, and modernization. The emergence of cloud computing and the widespread adoption of x86-based virtualization have created pressure to transition to more flexible and cost-efficient models. Enterprises are now increasingly evaluating how to decommission legacy CDOM platforms and migrate workloads to public or hybrid cloud infrastructures that offer elasticity, automation, and reduced operational costs.

1.2 Objectives of the Review

This review aims to explore strategies for architecting cost-effective transitions from CDOM-based infrastructures to cloud-based virtual machines. It investigates both the technical and economic dimensions of this migration, offering a structured methodology that includes architecture mapping, tool selection, image conversion, and infrastructure-as-code practices. Key challenges such as hypervisor compatibility, resource abstraction, and licensing portability are examined, along with approaches to cost reduction through instance sizing, storage tiering, and network optimization. The article further discusses automation pipelines, post-migration observability, and compliance enforcement. Real-world case studies from industries actively migrating CDOMs to cloud platforms are provided to contextualize the discussion. Ultimately, this review provides a blueprint for modernizing legacy virtualization environments into scalable and efficient cloud-native architectures.

2. CDOM Architecture Overview

2.1 Structure and Function

CDOMs, or Control Domains, are the primary domain types within Oracle's SPARC virtualization model that manage the physical resources of a server and act as the hypervisor's control point. They are responsible for the allocation and configuration of virtual CPUs, memory, I/O devices, and virtual disks to guest Logical Domains (LDOMs). Typically, the CDOM also runs virtual disk services (vds), virtual console services (vcc), and virtual switch services (vsw), which facilitate interaction between physical resources and guest domains. This setup enables a high degree of isolation and security for individual LDOMs, while centralizing control in a single domain. CDOMs are tightly integrated with the underlying SPARC hardware, and their management is achieved through tools such as ldm and virtinfo, often supplemented by Oracle Enterprise Manager Ops Center for orchestration.

2.2 Limitations in a Modern Context

Despite their robustness, CDOM-based architectures face several limitations when evaluated against today's cloud-native and hybrid cloud models. First, the requirement for physical SPARC hardware introduces a dependency that restricts elasticity and horizontal scaling. This makes capacity planning rigid and costly. Additionally, the lack of integration with modern CI/CD tools and Infrastructure-as-Code frameworks reduces the agility needed in DevOps-centric environments. The proprietary nature of CDOM virtualization also poses challenges for interoperability and toolchain extensibility. Licensing rigidity—especially Oracle's core-based models—further contributes to TCO inflation. Finally, CDOMs

do not easily support ephemeral or containerized workloads, which are increasingly preferred for microservices and agile delivery pipelines. These limitations highlight the need for a strategic migration to more agile and scalable cloud VM platforms.

3. Cloud VM Target Environments

3.1 Cloud Platforms and VM Profiles

When transitioning from CDOM environments, selecting a suitable cloud platform is a critical architectural decision. Oracle Cloud Infrastructure (OCI) provides the most direct path due to native support for Solaris and Oracle workloads, including SPARC-based VM shapes via Bare Metal or Dedicated Hosts. OCI's compatibility with Oracle Solaris 11 and integrated ZFS support makes it attractive for enterprises seeking minimal refactoring. AWS offers EC2 instances where x86-compatible Solaris x86 images can be deployed using custom AMIs or partner solutions. Azure also supports Solaris x86 through custom VM images, though its support may be more limited and often requires advanced image preparation. The choice of VM profiles ranging from burstable instances for dev/test to high-memory compute instances for production is driven by workload characteristics, licensing requirements, and cost targets.

3.2 Compatibility and Virtualization Models

Hypervisor compatibility is a significant technical factor in CDOM-to-cloud transitions. CDOMs rely on the SPARC hypervisor, while most public clouds operate on KVM or Xen. Solaris x86 images are generally more portable than SPARC binaries, which are not natively supported in most cloud environments. This creates a gap for SPARC workloads, often requiring emulation or partial replatforming. OCI offers the closest alternative with its support for SPARC workloads, but even here, care must be taken to align kernel versions and firmware expectations. For x86-based Solaris workloads, tools like virt-v2v can assist in converting images into formats suitable for KVM-based hypervisors. Furthermore, platform-specific drivers, metadata services, and network stack differences must be reconciled to ensure smooth guest operation. Successful transitions often involve adapting bootloaders, modifying GRUB configurations, and injecting cloud-compatible drivers.

4. Migration Strategy Design

4.1 Lift-and-Shift vs Replatforming

When planning a CDOM to cloud VM migration, two principal approaches emerge: lift-and-shift and replatforming. Lift-and-shift involves migrating the existing OS image, configurations, and applications with minimal changes. This strategy is suitable for organizations seeking rapid transition with reduced upfront engineering effort. It is often achieved using image conversion tools or disk cloning followed by deployment on a compatible cloud hypervisor. However, this method may carry over legacy inefficiencies and dependencies. On the other hand, replatforming involves rebuilding the application and its environment on the target cloud platform. While more time-consuming, it allows optimization for cloud-native services, including autoscaling, managed databases, and containerization. Replatforming is ideal for long-term modernization but requires careful dependency mapping and testing. The decision between the two strategies must balance technical debt, downtime tolerance, and modernization goals.

4.2 Target State Architecture Mapping

To effectively transition from CDOM to cloud, it is critical to map legacy components to their cloud-native counterparts. In a CDOM-based setup, roles such as control domain, virtual service domain, and I/O domain are often handled by a single host. In the cloud, these responsibilities are decomposed into discrete services. For instance, compute instances replace guest LDOMs, cloud block storage substitutes for virtual disks, and metadata services offer dynamic configuration akin to virtual device services. IAM roles and cloud-native access policies replace CDOM-based user and role management, while software-defined networking (SDN) takes the place of virtual switches and bridges. Understanding this mapping enables architects to redesign the environment with a focus on modularity, scalability, and cost-

efficiency, while maintaining service equivalence and security posture.

5. Cost Optimization Framework

5.1 TCO Analysis and ROI Planning

One of the key motivations for transitioning to cloud infrastructure is the potential for cost savings. A thorough Total Cost of Ownership (TCO) analysis should compare capital expenditures (CAPEX) in legacy CDOM environments—such as SPARC hardware maintenance, licensing, and cooling costs—against operational expenditures (OPEX) in the cloud. Cloud models allow for elastic resource consumption, converting fixed costs into variable costs. Reserved instances can significantly reduce compute expenses, especially for stable workloads. Organizations should also account for return on investment (ROI) timelines, which may vary depending on migration effort and platform selection. Additional financial levers include enterprise agreements, hybrid benefit programs, and negotiated discounts for long-term commitments.

5.2 Storage and Bandwidth Efficiency

Storage and bandwidth optimization are critical in reducing ongoing cloud costs. CDOM environments typically use ZFS or UFS on physical disks, which must be migrated to cloud storage formats. Cloud platforms offer various storage tiers (standard, high-performance, archive) that can be aligned with workload requirements. Techniques such as compression, deduplication, and sparse provisioning reduce both migration and runtime costs. Network egress often charged by volume can be minimized by localizing data access within availability zones and using caching layers. Additionally, snapshot-based backups and lifecycle policies can help manage long-term data retention without inflating costs.

5.3 License Portability and Consolidation

Licensing is a significant cost component when migrating from Oracle Solaris environments. Most SPARC-based CDOM deployments involve complex licensing arrangements, including per-core or per-socket models. Many cloud providers support Bring Your Own License (BYOL) policies, allowing organizations to port existing Oracle licenses to cloud infrastructure, particularly in OCI. License consolidation—running multiple apps or services on a single VM using containers or namespaces—can also improve utilization. Automated license tracking tools help ensure compliance while optimizing costs. Careful license reassignment and audit readiness are essential to avoid violations and unexpected charges.

6. Automation and Tooling

6.1 Image Conversion and VM Deployment

The initial step in migration involves converting legacy CDOM-based Solaris images into formats compatible with the target cloud hypervisor. Tools such as virt-v2v, Oracle's ovm_to_oci, and manual disk image exports allow administrators to extract and transform virtual machines. When using x86-based Solaris, the conversion process may be relatively straightforward, whereas SPARC workloads often require replication to supported OCI instances or even architectural emulation. After conversion, cloud APIs or orchestration platforms can deploy the images as new VMs, complete with metadata injection for networking, SSH keys, and startup scripts.

6.2 IaC and Orchestration

Infrastructure as Code (IaC) is a foundational principle in modern cloud operations. Tools like Terraform and Oracle Resource Manager allow the declarative provisioning of VM instances, networks, storage, and security policies. Ansible playbooks can then be used to automate configuration management and post-deployment operations. Combining IaC with version control (e.g., GitOps) ensures consistency, repeatability, and auditability of infrastructure changes. These tools also facilitate scaling and rollback, which are essential in phased migrations from CDOM environments.

6.3 Monitoring and Post-Migration Automation

After migration, maintaining visibility into system health and performance is essential. Tools such as Prometheus, Nagios, and OCI Monitoring can be configured to track resource utilization, alert on anomalies, and ensure SLA adherence. Automated health checks, including service pings, disk space audits, and memory usage thresholds, help prevent outages. When integrated with Ansible or scripting engines, these checks can trigger remediation actions—such as VM restarts, scaling operations, or configuration rollbacks. Monitoring dashboards and log aggregation through platforms like ELK or Grafana enhance operational transparency and troubleshooting.

7. Networking and Connectivity Design

7.1 Replacing Virtual Switches and Bridges

In CDOM environments, virtual networking is orchestrated using virtual switches (vsw), virtual NICs (vnet), and physical I/O domains that provide controlled access to external networks. In the cloud, these constructs are abstracted into software-defined networking (SDN) components such as Virtual Cloud Networks (VCNs) in Oracle Cloud Infrastructure, Virtual Private Clouds (VPCs) in AWS, and Virtual Networks (VNets) in Azure. These cloud-native equivalents allow the creation of subnets, route tables, and network security groups to manage east-west and north-south traffic. To map CDOM-based networking into the cloud, administrators must redesign address plans, translate physical-to-virtual bridge relationships, and define routing policies to emulate existing segmentation and isolation. Load balancing—previously handled within the CDOM or by third-party appliances—can be replaced by managed services such as OCI Load Balancer, AWS ELB, or Azure Load Balancer.

7.2 Hybrid and Multi-Cloud Connectivity

During phased migration or in hybrid scenarios, connectivity between on-premises CDOM infrastructure and cloud VMs is crucial. This can be accomplished through secure VPN tunnels, dedicated interconnects like Oracle FastConnect, AWS Direct Connect, or Azure ExpressRoute, and encrypted overlay networks. These links ensure data integrity and low latency during workload cutover, testing, or real-time synchronization. Proper DNS design and route advertisement are essential for seamless connectivity and failover. Cloud NAT gateways and bastion hosts can be configured to provide controlled access to private subnets, while maintaining a strong security posture. In multi-cloud environments, transit gateways and inter-cloud peering are leveraged to enable cross-provider communication and workload distribution, which is especially useful in active-active or failover architectures.

8. Storage and Backup Considerations

8.1 Virtual Disks and Storage Abstraction

In traditional CDOM setups, virtual disks (vdisks) are presented to LDOMs through services like vds backed by ZFS or UFS filesystems. Migrating these volumes to cloud platforms involves translating them into virtual block storage such as OCI Block Volumes, AWS EBS, or Azure Managed Disks. These services offer different performance tiers (standard, balanced, high-performance) that can be matched with application needs. Storage abstraction in the cloud allows dynamic resizing, snapshot creation, and cross-zone replication. While ZFS offers powerful snapshot and compression capabilities, these must be mapped or reconstructed using cloud-native equivalents or third-party agents. It is critical to assess IOPS requirements, latency sensitivity, and backup frequency when selecting storage configurations for cloud VMs.

8.2 Backup, Snapshots, and Replication

A robust backup strategy is essential during and after migration. While CDOM environments often rely on ZFS snapshots or custom backup scripts, cloud environments offer more scalable and automated options. Native snapshot functionality in OCI, AWS, and Azure can be scheduled and retained with lifecycle policies, enabling rapid rollback and point-in-time recovery. Tools like Commvault and Veeam

integrate well with cloud APIs to deliver advanced backup and recovery capabilities. Replication across availability domains or regions can be configured to support high availability and disaster recovery strategies. Additionally, object storage services (e.g., Oracle Object Storage, Amazon S3) can be used to archive backups in a cost-efficient manner with encryption and retention policies.

9. Security and Compliance Alignment

9.1 Identity and Access Management

CDOM-based security is often tied to local user accounts, role-based access within the Solaris OS, and administrative tools like RBAC and SMF. In cloud environments, security is driven by Identity and Access Management (IAM) frameworks that define who can access which resources and under what conditions. IAM roles and policies replace OS-level access controls, enabling granular permissions across compute, storage, networking, and orchestration layers. SSH key-based access must be tightly managed using secrets vaults or services like OCI Vault, AWS Secrets Manager, or Azure Key Vault. Implementing least-privilege access and auditing all administrative activities becomes a foundational requirement for securing cloud-hosted former CDOM workloads.

9.2 Compliance Standards Mapping

Compliance obligations such as PCI-DSS, HIPAA, ISO 27001, or FedRAMP are often more rigorously enforced in cloud environments. Mapping existing CDOM security controls—such as audit logs, access controls, and patching routines to cloud-native services ensures continuity of compliance. For example, OCI Audit or AWS CloudTrail can replace manual audit log collection. Tools like Oracle OS Management, AWS Systems Manager, or Azure Policy can automate patching and enforce policy compliance. Furthermore, configurations can be continuously evaluated using tools like SCAP or OpenSCAP to align with regulatory benchmarks. Automating these assessments and remediations enhances overall security posture while reducing manual overhead.

10. Real-World Case Studies

10.1 Manufacturing Sector: Minimal Downtime Transition

A global manufacturing company running Oracle ERP on LDOMs within CDOM environments needed to modernize its infrastructure without interrupting plant operations. The migration strategy involved replicating ZFS datasets to Oracle Cloud Infrastructure (OCI) using zfs send/receive, followed by snapshot-based synchronization. Preconfigured target VMs in OCI were deployed with identical memory, vCPU, and block storage profiles to the source LDOMs. Networking was redesigned using OCI's Virtual Cloud Network (VCN), and VPN connectivity enabled secure hybrid operations during cutover. The final switch-over occurred during a planned maintenance window, ensuring under one hour of downtime. Post-migration monitoring and patch automation were integrated using Oracle OS Management and OCI Monitoring. This case highlighted how phased migration and use of cloud-native equivalents for storage and networking enabled operational continuity with minimal disruption.

10.2 Financial Services: Cost-Efficient Refactoring

A leading financial institution, previously reliant on SPARC-based trading systems, opted for a phased replatforming strategy targeting cost optimization and cloud-native agility. The organization transitioned from CDOM-hosted LDOMs to custom-built x86 Solaris VMs on AWS EC2. Reserved instances were purchased based on projected workload demands, and Terraform modules were developed to ensure reproducible infrastructure builds. Stateless services were containerized using Docker and deployed on Amazon ECS, while stateful components were migrated to AWS EBS-backed EC2 instances. Compliance was enforced through integration with AWS Config and GuardDuty. The migration reduced infrastructure costs by over 40% and allowed integration into CI/CD pipelines for future enhancements, demonstrating the long-term value of rearchitecting legacy workloads.

10.3 Government: Security-Conscious Lift-and-Shift

A government agency with air-gapped CDOM environments faced regulatory pressure to adopt a FedRAMP-compliant cloud solution. The team conducted a full offline image export of Solaris LDOMs using ZFS snapshots, which were then transferred via encrypted physical media to a secure OCI tenancy. A bastion host was configured for administrative access, and isolated subnets were used to replicate the agency's segmented network design. IAM policies, encryption at rest, and audit logging were implemented to meet strict compliance requirements. Post-migration validation included vulnerability scans and policy audits using OpenSCAP. The project demonstrated that even highly restricted CDOM environments can be successfully transitioned using secure and compliant methodologies tailored for regulated industries.

11. Challenges and Mitigation Strategies

11.1 Legacy OS Constraints and Unsupported Drivers

One of the primary obstacles in CDOM-to-cloud migrations is the presence of legacy Solaris versions and hardware-dependent drivers. Older SPARC-based workloads may not be compatible with cloud hypervisors, and driver support for specific NICs or HBAs is often missing. In such cases, mitigation involves either upgrading the OS to a supported release or refactoring the application to run on Solaris x86 where available. In rare cases, emulation via QEMU or virtualization on OCI's SPARC bare-metal machines is used to preserve legacy workloads temporarily until a complete transition is feasible.

11.2 Data Consistency and Integrity Risks

Ensuring data consistency during migration especially for transactional workloads—is a critical concern. In CDOM environments, data is often tied to ZFS volumes or UFS filesystems. Without careful planning, inconsistencies may arise during snapshot transfers or VM replication. To mitigate this, administrators must quiesce applications before taking snapshots and validate post-migration data integrity using checksums or database-level verification. Technologies such as application-aware backup, logical replication (e.g., Oracle Data Guard), or real-time rsync with verification can help reduce risks during the migration window.

11.3 Staff Skills and Automation Complexity

Transitioning from CDOM to cloud environments requires a shift in operational paradigms, including IaC, automation pipelines, and cloud-native networking. Organizations may face skill gaps in areas such as Terraform scripting, cloud security policies, and CI/CD integration. To overcome this, phased adoption and cross-training programs are recommended. Utilizing managed services, consulting partners, or hybrid orchestration platforms like Ansible Automation Platform can reduce complexity. Embedding automation workflows early in the migration design also prevents manual errors and promotes consistent post-migration operations.

12. Future Outlook

12.1 Emulation and Containerization of Solaris Workloads

As cloud adoption matures, many organizations are evaluating alternatives to full VM-based migration. For non-performance-critical workloads, SPARC emulators like QEMU offer a temporary bridge to run legacy applications without reengineering. Containerization is also gaining traction, particularly for Solaris x86 workloads that can be isolated using Docker or Podman. While container support for Solaris is limited, initiatives like Zones-based containers or Solaris-branded zones provide lightweight alternatives for specific use cases.

12.2 AI-Driven Cost Optimization and Resource Tuning

AI and ML tools are increasingly being adopted to optimize cloud resource usage. Cloud providers offer native services such as AWS Compute Optimizer, Oracle Cloud Advisor, and Azure Cost Management

that analyze usage patterns and recommend resizing or scaling strategies. Integrating these insights with Terraform or Ansible workflows enables dynamic rightsizing, automated schedule-based scaling, and predictive failure remediation. These tools help reduce waste and ensure that cloud resources align with actual workload demands.

12.3 Serverless Alternatives for Stateless CDOM Workloads

Stateless services previously hosted in CDOMs such as logging agents, schedulers, or lightweight monitoring tools can often be replaced with serverless architectures. Function-as-a-Service (FaaS) platforms like AWS Lambda, OCI Functions, or Azure Functions offer scalable, event-driven execution with minimal overhead. These services reduce infrastructure management complexity and can be integrated into pipelines for provisioning, remediation, and alerting. For certain use cases, moving from VMs to managed databases, message queues, and object storage may further simplify the infrastructure stack.

13. Conclusion

The transition from traditional Control Domains (CDOMs) to cloud-based virtual machines marks a pivotal evolution in enterprise infrastructure strategy. Legacy CDOM environments, though robust and stable, are inherently bound by hardware constraints, static resource allocation, and limited integration with modern DevOps practices. Migrating to cloud VMs whether via lift-and-shift or through replatforming—offers the promise of agility, scalability, and cost-effectiveness. However, achieving a successful migration requires a thoughtful approach to architecture mapping, automation tooling, compliance alignment, and financial modeling.

This review has examined the structural differences between CDOMs and cloud-native virtual environments, highlighting the technical considerations in virtualization compatibility, storage abstraction, and networking design. It also addressed how infrastructure-as-code (IaC), monitoring frameworks, and post-migration orchestration tools can automate the entire VM lifecycle, ensuring consistency and resilience. Moreover, the importance of cost optimization—through licensing strategies, storage tiering, and right-sizing has been underscored as a key driver of sustainable operations post-migration. Real-world case studies from manufacturing, financial services, and government sectors demonstrated how phased migration, cloud-native integration, and stringent compliance controls can be tailored to sector-specific demands. Despite inherent challenges such as legacy OS support, skill gaps, and data integrity risks organizations can mitigate these through structured planning, cross-functional collaboration, and the adoption of hybrid connectivity models.

14. Reference

1. Xu, X., & Diaz-Mercado, Y. (2019). Multi-Agent Control Using Coverage Over Time-Varying Domains. *2020 American Control Conference (ACC)*, 2030-2035.
2. Orgeron, O. (2008). An Introduction to Logical Domains, Part 4. *login Usenix Mag., 33*.
3. Wang, L., Li, Q., & Zhou, X. (2021). Continuous L-domains in logical form. *Ann. Pure Appl. Log., 172*, 102993.
4. Battula, V. (2021). Dynamic resource allocation in Solaris/Linux hybrid environments using real-time monitoring and AI-based load balancing. International Journal of Engineering Technology Research & Management, 5 (11), 81–89. https://ijetrm.com/
5. Benzekki, K., Fergougui, A.E., & El Alaoui, A.E. (2016). Software-defined networking (SDN): a survey. *Secur. Commun. Networks, 9*, 5803-5833.
6. Vasudevan, A. (2019). The Uber eXtensible Micro-Hypervisor Framework (uberXMHF). *Practical Security Properties on Commodity Computing Platforms*.
7. Vilarinho, E.C., Bonde, A., Muzio, A., Russinovich, M., Fontoura, M., & Bianchini, R. (2017).

Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms. *Proceedings of the 26th Symposium on Operating Systems Principles*.

8. Ambati, P., Goiri, Í., Frujeri, F.V., Gun, A., Wang, K., Dolan, B., Corell, B., Pasupuleti, S., Moscibroda, T., Elnikety, S., Fontoura, M., & Bianchini, R. (2020). Providing SLOs for Resource-Harvesting VMs in Cloud Platforms. *USENIX Symposium on Operating Systems Design and Implementation*.
9. Xie, J., Yu, F., Huang, T., Xie, R., Liu, J., Wang, C., & Liu, Y. (2019). A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges. *IEEE Communications Surveys & Tutorials, 21*, 393-430.
10. Yan, Q., Yu, F., Gong, Q., & Li, J. (2016). Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges. *IEEE Communications Surveys & Tutorials, 18*, 602-622.
11. Soh, J., Copeland, M., Puca, A., & Harris, M. (2020). Infrastructure as Code (IaC). *Microsoft Azure*.
12. Chinamanagonda, S. (2019). Automating Infrastructure with Infrastructure as Code (IaC). *International Journal of Science and Research (IJSR)*.
13. Oinam, A.C., & Nongbri, J.P. (2016). Software Migration from LibSys 4.0 to Koha: Post Migration Scenario in a University Library of North-Eastern Hill University, Shillong, Meghalaya, India.
14. Krishnan, A., Swarna, S., & S, B.H. (2020). Robotics, IoT, and AI in the Automation of Agricultural Industry: A Review. *2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC)*, 1-6.
15. Bueno, G., Saquetti, M., Azambuja, J.R., & Cordeiro, W. (2020). Defending lightweight virtual switches from cross-app poisoning attacks with vIFC. *Proceedings of the SIGCOMM '20 Poster and Demo Sessions*.
16. Battula, V. (2022). Legacy systems, modern solutions: A roadmap for UNIX administrators. Royal book publishers
17. Wang, Y., Gobriel, S., Wang, R., Tai, T., & Dumitrescu, C. (2018). Hash Table Design and Optimization for Software Virtual Switches. *Proceedings of the 2018 Afternoon Workshop on Kernel Bypassing Networks*.
18. Oppegaard, A., & Christoffer, M. (2009). Evaluation of performance and space utilisation when using snapshots in the ZFS and Hammer file systems.
19. Kofahi, I., & Alryalat, H. (2017). Enterprise Resource Planning (ERP) Implementation Approaches and the Performance of Procure-to-Pay Business Processes: (Field Study in Companies that Implement Oracle ERP in Jordan). *Int. J. Inf. Technol. Proj. Manag., 8*, 55-71.
20. Mohr, R., & Howard, A.P. (2019). Provisioning ZFS Pools On Lustre. *Practice and Experience in Advanced Research Computing 2019: Rise of the Machines (learning)*.
21. Kumar, L., Rao, M.K., & Viswanadham, P. (2014). Addressing the Issues and Challenges in Implementation of Ias & Ifrs With Oracle Erp. *IOSR Journal of Economics and Finance, 4*, 45-49.
22. Valentino, E.D., Melchiorri, A., & Silk, J.I. (2019). Cosmological constraints in extended parameter space from the Planck 2018 Legacy release. *Journal of Cosmology and Astroparticle Physics, 2020*, 013 - 013.