International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(1s) 2026 pp. 1-8
https://doi.org/10.70917/ijcisim-2026-2036
Article

The Zero-Touch Data Center: Lights-Out Operations at Scale

Hardik Mahant[1], Ajay Prasad[2], Rohit Kumar Shaw[3]

[1] San Jose, California (CA), USA. hardik.s.mahant@gmail.com
ORCID: 0009-0009-9374-911X
[2] Fremont, California (CA), USA. rushtoajay@gmail.com
ORCID: 0009-0003-4065-0833
[3]Plano, Texas (TX), USA. rohitshaw.infosec@gmail.com
ORCID: 0009-0009-3751-0809

Abstract: Research shows that over 90% of the critical incidents that occur in hyperscale datacenters are related to same event type and with potentially same hardware a.k.a. host. In this paper, its described a Zero Touch Datacenter framework, Self Healing Architecture (SHArch) that takes away a majority chunk of work from the IT Operations team, responsible to maintain hyperscale datacenters. The framework orchestrates and heals these events and provides a consolidation process to further analyze noisy hosts for a permanent solution to these recurring events. SHArch has been extended to support IBM z/OS mainframe environments, enabling true lights-out operations across hybrid x86 and mainframe fleets. It achieves higher efficiency, brings end to end visibility in the entire operations process in real time. It is truly autonomous as it scales, remains highly available and remains flexible for adopting to continuously changing fault patterns.

Keywords: Zero-Touch, Site Reliability, Lights-Out Operations, Self-Healing, Self-Remediation, Infrastructure Management, Data Center Operations

1. Introduction

In a typical operational environment, a team of site reliability and system administrators is responsible for monitoring infrastructure typically in a multi-region data center or highly available cloud infrastructure setup in a 24x7 on-call setup. The action is taken after the critical event has occurred. These critical events typically result in business outages costing millions, often billions of dollars to the business or a service[1]. If handled manually, it may take hours for troubleshooting the incident and identifying the solution. Often the issue spans across multiple layers of the infrastructure and may need longer time to identify and resolve the incident. We analyzed over 200,000 incidents for about 3 months of data and found that over 90% of the incidents that occurred had occurred previously, also identified as “known issues”. SHArch is also designed to support IBM z/OS environments, where its robust Parallel Sysplex, automatic restart capabilities, and mature automation tools (such as GDPS and System Automation) align naturally with zero-touch remediation principles. If an incident is known, the operations team usually have a runbook that can be applied to resolve the issue at a given layer and this applies across multiple layers of the infrastructure. In this paper, we are implementing Self Healing Architecture (SHArch) which can reduce the resolution time of an incident by 90% if it is a known issue and a runbook is readily available to remediate. This is a significant improvement on how the operations and site reliability teams operate and help in saving the cost of operations and business impact due to the outage. SHArch can integrate with a wide variety of open source and proprietary monitoring systems to receive the critical event signals. SHArch is distributed, scalable, highly available and reliable system which if deployed in a mission critical infrastructure would save billions of dollars every year and help the data center infrastructure achieve the highest reliability on the quality of service.

2. Literature Review

With a constant growth of data, the operations become increasingly complex. To handle ever growing data, a hyperscale infrastructure is needed and to maintain this infrastructure, it's crucial to strategically develop and leverage automated handling of the infrastructure operations [5]. Realtime data processing for handling the incoming alerts is challenging with traditional log based systems and the applied ML models don't seem to accurately provide a desired result in identifying and achieving the remediation [6]. Some studies have been conducted on managing isolated layers of a hyperscale infrastructure, employing multiple hybrid data centers. While this can be effective at the layer at which it is applied, the challenges remain on the other layers. Without addressing a wholistic solution, it is not possible to achieve a true Zero Touch Datacenter. Securely implementing and deploying these solutions at a single layer is also a challenge due to the complexity of the subject area [7]. While automated maintenance is a key area to be addressed for higher cost savings, allocating resources and power is also an area that needs similar solutions to be applied [8]. Proposed architecture can be applied to multiple domain areas including application, hardware, networking, server management, resource management and power management by adding the runbooks for the respective domain areas and building the rules for handling events for the respective domains. The framework further extends to z/OS mainframe systems by leveraging z/OS-specific runbooks that interact with facilities such as z/OS System Automation (SA), RMF (Resource Measurement Facility), SMF records, and console message automation for event-driven self-healing.

Some studies highlight the importance of learning from historical data and applying Machine Learning techniques for predictive infrastructure maintenance. We address this area in our future research section. This study indicates that the older ways to identify and mitigate anomalies require significant time to troubleshoot and fix the faults. It poses challenge as the data is growing astronomically and the traditional methods would not be able to keep up with the scale, contributing towards increased downtime and business loss, suggesting the need to employ various ML techniques in their solutions [9]. Automation improves infrastructure by reducing manual efforts. With complex and heterogeneous infrastructure, there is a huge need to automate not only provisioning but also monitoring and maintenance of the infrastructure. Identifying and resolving issues before they impact users and business is very crucial and automating these operations is the key to running successful operation management, by bringing efficiency, reliability and stability [10]. Traditional maintenance methods require planned downtime and resource re-allocation, often provisioning parallel infrastructure before starting the maintenance on the existing systems. Systems that analyze historical data and accurately predict a potential failure are extremely complex to implement and require deep subject matter expertise [11]. Some studies have been conducted on just looking at the data based on hypervisors for predicting application failures. This approach has a limitation as the issue may not be at hypervisor level, especially if it is a shared tenant model where multiple users are hosted. This model is known for its nature to be hostile and make other tenants suffer through poor performance and eventually cause increased latency or component failures. The research has a limitation of focusing on one area to predict the failures while the potential problems could be in the surrounding layers [12]. Studies also indicate the need to move from reactive to predictive systems for outage prevention. While this may bring significant improvement in overall maintenance of the infrastructure, it may be complex to constantly identify and train data for various use cases. The prediction is different for different metrics of the systems and the algorithms need to change by each component to achieve desired outcomes. While there was a noted success for certain areas, the study shows that it cannot be applied confidently across multiple layers of the stack. The varying nature of the data requires it to be used and tweaked differently for better outcomes [13]. Consequences of outages and mismanagement costs staggering amount of money. Outages result in lost productivity and waste power resources as an outage in one layer can cause other layers to be idle. The research proposes the use of AIOps by learning massive amounts of data for optimizing data center improvements. The investment in AIOps costs substantial amount of money but would gain profitability in a long term as it matures and takes on new tasks by learning about the environment [14]. The applicability of the automated monitoring and maintenance is not limited to compute infrastructure in data centers but it also applies to public systems like railway systems, traffic systems, power grids. It's equivalent in terms of monitoring large amount of data and the ability to predict and prevent outages before they happen. Reactive and predictive approaches are usually combined to improve the overall health of the system and improve the reliability of the service [15].

3. Methodology

We introduce Self Healing Architecture (SHArch) in this paper which aims to heal faults in data center scale, on-premises and cloud based hyperscale infrastructures. SHArch can handle millions of events per second as it's very dynamic in terms of scaling horizontally [2] to handle bursts of event floods. The event flood occurs when more than one racks hosting multiple servers encounters an outage situation due to network or system fault. This

phenomenon can create 1000s of events per minute and would need a parent level issue identification and fix. SHArch scales up itself to handle these events and receive the event payloads. Figure 1 shows SHArch's system architecture diagram. We describe the architecture of SHArch in 12 Steps.

Step 1 Infrastructure Configuration

In this step, we define the configuration of data centers. The data centers are combination of on-premises physical locations where the machines are deployed to serve application loads and the ones defined in public cloud providers where the infrastructure can be deployed in multiple regions with auto-scaling rules [3]. The infrastructure metadata is stored in a Configuration Management Database (CMDB) where the logical information of available servers is maintained. This CMDB contains a golden copy of the fleet managed in the data center. Once configured, it enables the discovery of this infrastructure to be discovered in monitoring system. Monitoring systems contain the configuration rules that need to be deployed for observability on each server. Table 1 shows the critical metrics that covers over 80% of the incidents that occur at system and hardware layer of the infrastructure. In Figure 1, the blocks representing various Rack locations represent a physical data center or a virtual private network representing a data center in cloud. Each block represents a geographical region where the data center or virtual private network is deployed. This represents a globally deployed infrastructure for a business service that is designed to serve billions of users and millions of transactions per minute. Hybrid configurations that include IBM z/OS mainframes (running on z16 or earlier hardware) are fully supported through CMDB entries and monitoring agents that consume z/OS console messages, SMF Type 90/92 records, and health checks.

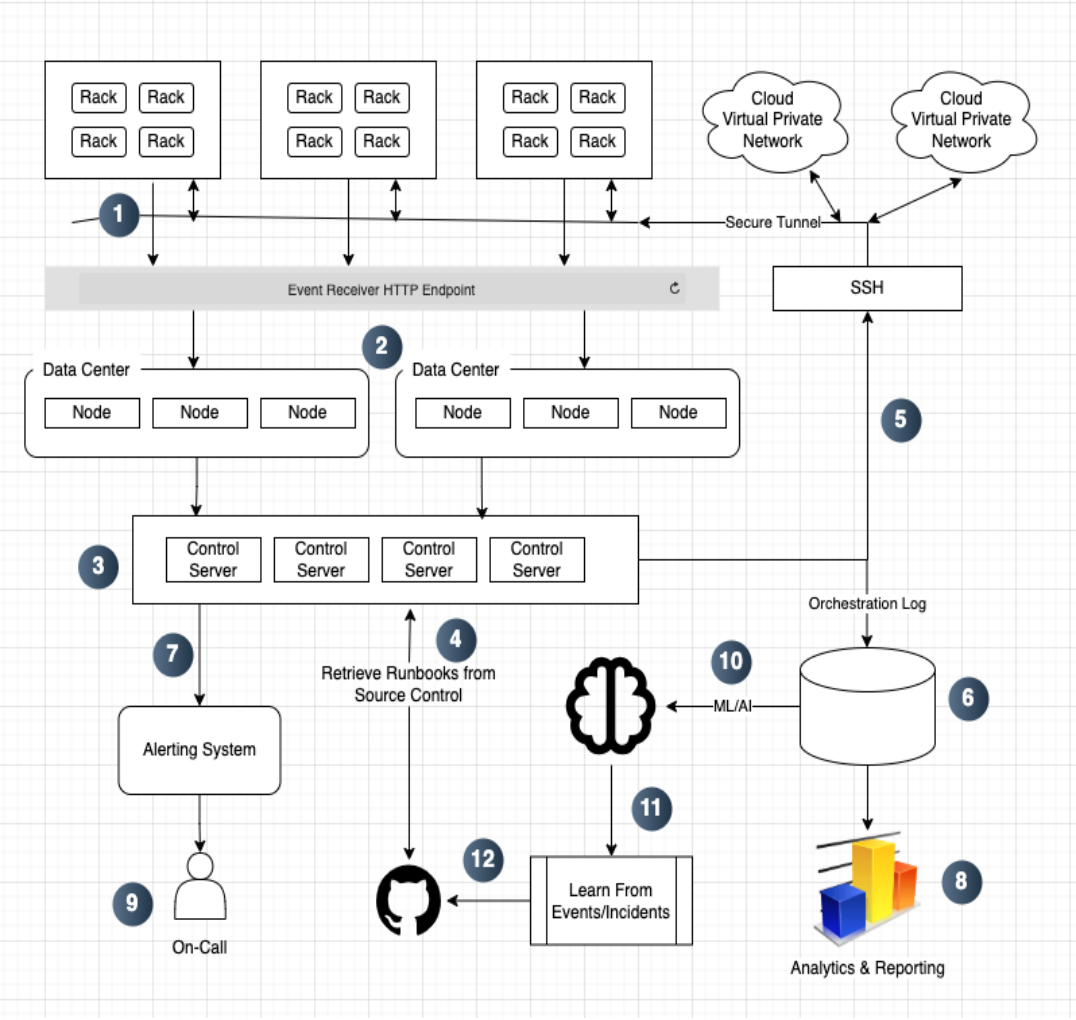

Figure 1 Self Healing Architecture: Event flow and processing

Step 2 Event Receiver Gateway

This is a very critical component which is responsible for ingesting all the critical events that are happening at the infrastructure. This layer can scale up as needed and shrinking back to the minimal required instances layer. It is distributed in multiple availability zones for high availability and fault tolerance. To handle the bursts of the events, it consists of a queuing layer which allows the control servers to process critical events by their priorities. Data center nodes are constantly receiving and processing the events from the queue.

Metric Name	Measurement	Critical Threshold or Fault Pattern
CPU Load Average	5, 10, 15 Minutes Load Average	>=.85 && <= .95
Physical memory Usage	Free Memory (RAM)	>=90%
Disk Usage	Free Disk Space	>=90%
Network Latency	Network Latency	Packet Loss or High Latency
System Process Count	Number of Acceptable Processes	>2000
System Uptime	Number of days it's been up	>6 Months
File System Usage	Verify all file system health	Partition > 90% High inode Usage
Kernel Memory	Kernel Memory (Root, Swap)	slabtop -o Slab /proc/meminfo
Kernel Parameters	Parameters on initial boot	/etc/sysctl.conf /etc/sysctl.d/*.conf sysctl -w (RT only, non persistent)
Kernel Panics	Kernel Fatal Errors	journalctl -k -b -1 | grep "panic"
Application Response Time	Against defined SLA2	Log and watch critical API response times
System Error Rate	/var/log/messages for Errors	systemctl or journalctl -u service_name
Application Error Rate	Errors in Application Logs	Check application logs in logs directory of the app
Container Metrics	Against allocated values	Top K containers by CPU Usage Container memory usage (Non Image) Container file system usage (Non Image)
Hypervisor Metrics	Overall virtualization health	High CPU Usage, Memory Usage, Host Memory Usage
Power Consumption	Rack Level Aggregation	iLO, IPMI or SNMP based PDU APIs
Network Connectivity	Check for Outages and Slowness	Setup 5s ICMP playbook for target and source hosts

NTP Drift	NTP Synchronization	Use chrony for sync

Table 1 Zero Touch Observability Metrics for Monitoring

Step 3 Orchestration Layer

This layer identifies the type of alert and the host on which the fault has occurred. It then identifies This layer has a set of nodes which are whitelisted to login to the remote machines across different data centers located in different time zones. This layer needs to be configured carefully due to its elevated privileges to login across the entire infrastructure. A routing table is configured which defines the IP Addresses of the machines that can communicate with these servers which secures these servers from being vulnerable to a wider internet traffic.

Step 4 Identify Runbook and Prepare to Orchestrate

In Step 3, the architecture discovered the host on which the fault occurred and the type of fault. In this step, it retrieves the runbook stored in a central repository of runbooks. A runbook is either an Ansible playbook [4] or a custom script that can isolate and heal the fault. The runbooks are maintained by operations engineering team and regularly updated to address new use cases related to changing infrastructure configurations.

Step 5 Access Faulty Server and Heal

Once the runbook has been identified, the flow then logs into the target host using Secure Shell (SSH) method on port 22. It then orchestrates the fault as per runbook workflow and brings the server back to a normal state.

Step 6 Collect Orchestration Logs and Create Audit Trace

Once the remediation is complete, the logs are collected at this stage and saved to a data lake for analytics and reporting purposes.

Step 7 Create a Ticket for Unsuccessful Remediation

In some scenarios, there could be failures at Step 5 and it needs to be handled by an on-call engineer. In this case, the architecture first logs the event and creates a ticket in the alerting system.

Step 8 Analytics and Reporting

As the critical events occur and remediated, the data becomes available in almost real time for review. This helps in designing a dynamic action plan depending on the criticality of the events.

Step 9 Engage On-call Team

The on-call team is engaged immediately for troubleshooting and resolution. The partially successful steps are available in the data warehouse and analytical dashboard for the responding engineering team to review and act on it. This saves enormous time as the fault classification has been made and initial investigation logs are already available, cutting down the resolution time by at least 80% in failed self-healing scenarios.

Step 10 Learn New Patterns

Historical data is used by Machine Learning and AI algorithms for identifying the patterns and updating the training dataset.

Step 11 Update Runbooks Based on New Learning

If new faults are discovered, the algorithm adopts the solution provided by an engineer and updates the runbooks as needed. This is a constant process of improvement where a single effort in remediation can be replicated and reused numerous times by SHArch.

Step 12 Update Run Books and Deploy to Central Repository

Finally, the new steps are reviewed and accepted by the operations team and source control system gets an updated with the new run book version.

4. Results

We analyzed data for over 200000 critical incidents for over 950 nodes and found that the remediation percentage was between 60-90% of the critical events that occurred. If we compare the remediation handled via SHArch with manual remediation, the time savings are astronomical. The remediation successfully handled 19 critical types of checks across application, network, system, kernel and hardware layers of the architecture. Early pilots on z/OS LPARs demonstrated similar success rates for common faults such as address space abends, coupling facility issues, JES2/3 spool shortages, and DASD space exhaustion. SHArch is a system that achieves a true Zero-Touch Data Center by achieving operational excellence and continuous adoption of new and improved runbooks. The results come from the data collected over the three months for a data center configuration. Achieving this benchmark provides a promising path forward for SHArch framework to be adopted at a broader scale.

The results in Table 2 show that a significant portion of the events generated were handled by the framework, requiring zero human involvement. Deploying similar in production environment could save thousands of human hours every month. Figure 2 visually represents a significant amount of incidents were handled and healed by SHArch.

Monitored Metric	Not Remediated	Remediated	Total Events
Application Error Rate	3622	9533	13155
Application Response Time	2483	10655	13138
Container Metrics	3375	9991	13366
CPU Load Average	1345	11822	13167
Disk Usage	3584	9573	13157
File System Usage	4651	8485	13136
High inode Usage	3764	9497	13261
Hypervisor Metrics	1912	11041	12953
Kernel Memory	2067	11284	13351
Kernel Panics	1359	11850	13209
Kernel Parameters	4724	8444	13168
Network Connectivity	4983	8117	13100
Network Latency	4869	8243	13112
NTP Drift	1792	11179	12971
Physical memory Usage	2621	10407	13028
Power Consumption	2178	11063	13241
System Error Rate	1953	11182	13135
System Process Count	3194	10006	13200
System Uptime	3953	9199	13152
Grand Total	58429	191571	250000

Table 2 Remediation Status for The Key Monitored Metrics

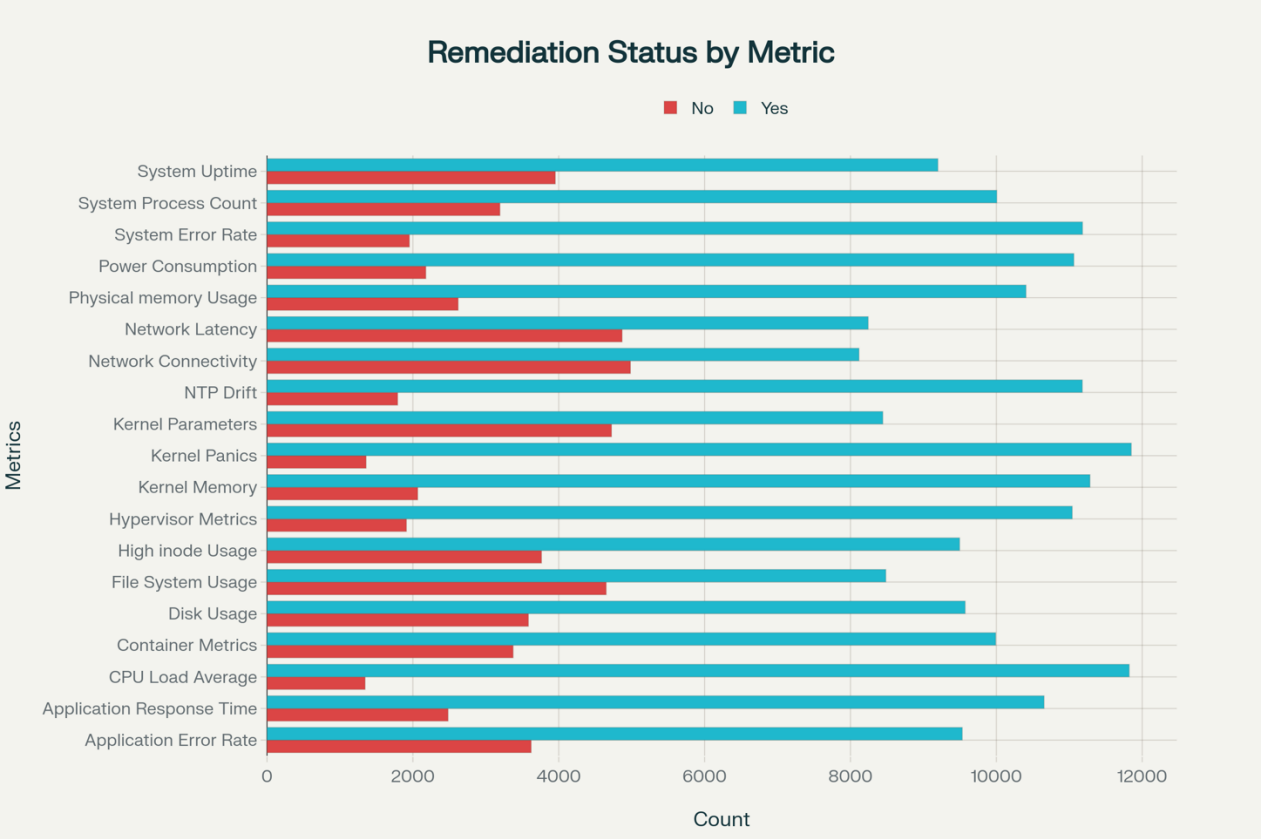

Figure 2 Visual Representation of Remediation Results

5. Future Research

As the next steps for SHArch framework, the native integration of AI intelligence would be an area that we would like to explore and expand on. There has been research on predictive analytics where the AI system would be predicting a critical event and try to isolate and remediate the event before it happens. This requires a very careful partnership with various subject matters to carefully monitor the data across different metrics and evaluate the performance. It is often challenging to correlate event across multiple layers and qualify those events as a single business impact. AI can be used expand in this area of the field and if implemented successfully, it'll help the organizations achieve higher efficiency by handling dynamic fault evaluation and automatic remediation. Advanced AI techniques like Model Context Protocol (MCP) can be integrated, including z/OS-specific predictive models using SMF data and RMF metrics, to provide more context towards the automated workflows and runbooks whereas reinforced learning methods can be used to train the Large Language Models (LLMs) on unseen scenarios and remediate those. These improvements would help address the limitations in a rule based orchestration.

6. Conclusion

We went through the SHArch framework for configuration based hyperscale hybrid datacenters. With over 90% of the critical events handled by SHArch, it achieves true Zero Touch approach for the common faults that occur daily in these datacenters. SHArch framework allows organizations maintain a small competent team to maintain multibillion dollar datacenter infrastructure hence it becomes extremely important to automate as many operations as possible. We also explored the possible avenues for leveraging AI for handling dynamic and unforeseen faults and predictively handle these failures before they cause any business impact.

References

1. Gadget Hacks. (2025, August 11). When Apple's Cloud Suddenly Vanishes: The Hidden Cost of Our Always-Connected Lives. Gadget Hacks. https://ios.gadgethacks.com/news/when-apples-cloud-suddenly-vanishes-the-hidden-cost-of-our-always-connected-lives/

2. Custer, C. (2023, April 10). Vertical vs. horizontal scaling: what's the difference and which is better? Cockroach Labs. https://www.cockroachlabs.com/blog/vertical-scaling-vs-horizontal-scaling/
3. Wikipedia Contributors. (2018, December 26). Autoscaling. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Autoscaling
4. Ansible playbooks — Ansible Documentation. (n.d.). Docs.ansible.com. https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_intro.html
5. Wu, B., Gong, Y., Zheng, H., Zhang, Y., Huang, J., & Xu, J. (2024). Enterprise cloud resource optimization and management based on cloud operations. Applied and Computational Engineering, 67(1), 8–14. https://doi.org/10.54254/2755-2721/67/20240667
6. Mallreddy, S. R., & Vasa, Y. (2023). Predictive maintenance in cloud computing and devops: ml models for anticipating and preventing system failures. IJRDO-Journal of Computer Science Engineering. https://doi.org/10.53555/cse.v9i8.6129
7. Friesen, M., Wisniewski, L., & Jasperneite, J. (2022). Machine Learning for Zero- Touch Management in Heterogeneous Industrial Networks - A Review. IEEE International Workshop on Factory Communication Systems, 1–8. https://doi.org/10.1109/WFCS53837.2022.9779183
8. Cong, P., Zhou, J., Wang, J., Wu, Z., & Hu, S. (2023). Learning-Based Cloud Server Configuration for Energy Minimization Under Reliability Constraint. IEEE Transactions on Reliability, 1–13. https://doi.org/10.1109/tr.2023.3234036
9. Leveraging Machine Learning for Predictive Maintenance in Cloud Infrastructure. (2024). International Research Journal Of Modernization In Engineering Technology And Science. https://doi.org/10.56726/irjmets61247
10. Bysani, V. (2024). Automation in Cloud Infrastructure Management: Enhancing Efficiency and Reliability. Indian Scientific Journal Of Research In Engineering And Management, 08(06), 1–5. https://doi.org/10.55041/ijsrem35750
11. Saini, N., Yadav, A. L., & Rahman, A. (2024). Cloud Based Predictive Maintenance System. 1–5. https://doi.org/10.1109/icrito61523.2024.10522398
12. Domingos, J., Cerveira, F., Barbosa, R., & Madeira, H. (2023). Predicting Cloud Applications Failures from Infrastructure Level Data. 9–16. https://doi.org/10.1109/dsn-w58399.2023.00023
13. Jain, M., Mutlu, B. O., Stam, C. J., Strube, J., Schupbach, B., John, J., & Pellico, W. (2025). AI-Enabled Operations at Fermi Complex: Multivariate Time Series Prediction for Outage Prediction and Diagnosis. https://doi.org/10.48550/arxiv.2501.01509
14. Joshi, S. P., Serebryakov, S., Nanjundaiah, D., & Hegde, T. (2024). AIOps and Sustainability: Transforming Data Centers for a Greener Future. 1867–1871. https://doi.org/10.1109/scw63240.2024.00234
15. Olufemi, O. D. (2024). AI-enhanced predictive maintenance systems for critical infrastructure: Cloud-native architectures approach. World Journal of Advanced Engineering Technology and Sciences. https://doi.org/10.30574/wjaets.2024.13.2.0552