Eduvest – Journal of Universal Studies
Volume 5 Number 8, August, 2025
p- ISSN 2775-3735- e-ISSN 2775-3727

Automating Server Deployment with Ansible to Improve Performance, Virtualization, and Network Security

Abdhul Nur Hidayat, Andi WRE
Universitas Atma Jaya Yogyakarta, Indonesia
Email: 245312937@students.uajy.ac.id, andi.emanuel@uajy.ac.id

ABSTRACT

Modern IT infrastructure management faces significant challenges in automation, virtualization, and network security. Manual server management processes often lead to inefficiencies and potential misconfigurations. This research aims to develop an automated server deployment method using Ansible, integrating container-based virtualization, and implementing network security policies with FortiGate. The methodology includes implementing Ansible automation scripts, testing virtual environments with Docker, and configuring FortiGate firewalls to enhance network security. Results demonstrate that Ansible automation significantly accelerates deployment processes, improves server resource management efficiency, and strengthens network security through more consistent and scalable firewall configurations. The integration of these technologies provides an efficient, flexible, and secure solution for modern IT infrastructure, addressing key challenges in automation, virtualization, and security while optimizing operational performance. The study concludes that automated approaches with Ansible, combined with proper virtualization and security measures, can substantially enhance IT infrastructure management. The implications of these research findings provide practical contributions in reducing reliance on manual processes, improving configuration consistency, and offering a replicable framework for automation implementation in the industrial and education sectors.

KEYWORDS Automation, Virtualization, Network Security, Ansible, FortiGate

INTRODUCTION

Computer networks play a crucial role in supporting business and organizational operations. Network availability and reliability are critical to maintaining productivity and operational efficiency (Islami, Musa, & Lamsani, 2020). However, the main challenge faced is how to secure data from unauthorized access or cyberattacks, while also ensuring system performance remains optimal. Manual processes in server and network management are often time-consuming, prone to misconfiguration, and inefficient at scale (Sun, 2022). In addition, the need for high service availability requires solutions that reduce the risk of system failure and increase automation in infrastructure management (Yalestia, Chandrawaty, & Hariyadi, 2021).

Network security is an important aspect of maintaining system integrity against external threats. One of the key components in network security is a firewall, which serves to block unauthorized access and cyberattacks (Khumaidi, 2021). However, relying on a single firewall can increase the risk of system failure (Egbuna, 2022; Hasbi et al., 2022; Sethuraman et al., 2023; Sharma et al., 2019; Singh et al., 2023; Thompson et al., 2023). Therefore, a more robust

network security solution is needed, such as the use of FortiGate firewalls, which provide multi-layered protection and better security management (Wicaksono & Widiasari, 2022).

In addition to security, reliable and efficient servers are also key factors in supporting the continuity of business operations and educational institutions (Hariyadi & Marzuki, 2020). Traditional server infrastructure often faces challenges in scalability and availability. Failover clustering technology addresses this by integrating several servers into one system, ensuring continued service even if one server fails. However, the implementation of this technology still requires a complex configuration process (Shimonski, 2003).

The implementation of virtualization and containerization technology is increasingly being used because it can reduce infrastructure costs and increase operational efficiency (Wijaya, Abdurrohim, Tugiyono, & Rumandan, 2023). With container-based virtualization, multiple services can run in isolation without interfering with each other on a single physical system (Pratama & Hariyadi, 2021). Kubernetes, as a container management platform, enables more efficient management of services at scale. However, many institutions are not yet fully prepared to migrate to this system, as they still face obstacles in implementation and maintenance (Putro & Supono, 2022).

Several previous studies have explored the use of automation and virtualization in IT infrastructure. Hariyadi & Marzuki (2020) implemented Ansible for configuration management of virtual private servers, demonstrating its effectiveness in reducing manual intervention. Islami, Musa, & Lamsani (2020) utilized Ansible for network automation to configure routing protocols on Cisco and Mikrotik routers, highlighting its flexibility. In the context of virtualization, Kristianto (2023) designed a server optimization system using VMware, while Putro & Supono (2022) discussed the challenges and opportunities of container-based virtualization with Kubernetes. However, these studies have not fully integrated automation, virtualization, and network security into a comprehensive framework. This research aims to fill that gap by combining Ansible automation, KVM virtualization, Kubernetes container management, and FortiGate security into a unified approach to improve server deployment, performance, and security (Arunawati, 2020; Kristianto, 2023).

To overcome the various problems mentioned above, this study aims to automate server management using Ansible. Ansible is an automation tool based on Infrastructure as Code (IaC) that allows server configuration and management to be performed faster, more efficiently, and consistently. By implementing Ansible, system deployment and update processes can be automated, reducing the risk of misconfiguration and accelerating response times to changing infrastructure needs. This research will also explore the integration of containerization technology with Kubernetes to improve the scalability and efficiency of application management (Putri et al., 2021). In addition, network security will be strengthened through the implementation of the FortiGate firewall to provide better protection against cyber threats (Rina & Ridha, 2021). With this approach, organizations are expected to manage their IT infrastructure more efficiently and securely.

This research aims to develop automation methods in deployment and server management using Ansible to improve operational efficiency. It also applies containerization and Kubernetes technology to improve the scalability and management of applications in a virtualized environment. Network security solutions with FortiGate firewalls are integrated to enhance protection against cyber threats. Furthermore, this study evaluates the effectiveness of automation methods in improving the reliability and security of modern IT infrastructure.

The various concepts and theories underpinning this research will be discussed, including Infrastructure as Code (IaC) with Ansible, which enables automatic and consistent management of servers; virtualization and containerization technologies that support the isolation of services on a single physical system; network security with the FortiGate firewall

that provides layered protection against cyberattacks; and container management with Kubernetes, which simplifies managing container-based applications at scale.

Through this research, it is hoped that solutions can be found to reduce complexity in server and network management through Ansible-based automation. Additionally, this study aims to improve the efficiency of system deployment and updates by achieving faster and more consistent processes, strengthening network security through FortiGate firewall implementation, and providing recommendations for organizations adopting automation technology to enhance the efficiency and security of their IT infrastructure. Thus, this research is expected to offer effective solutions applicable across various industrial and educational sectors to improve the performance, security, and availability of overall IT infrastructure.

METHOD

This research employs a design and implementation approach combined with experimental testing to evaluate the effectiveness of Ansible-based automation in server deployment, virtualization, and network security. This research uses the Network Development Life Cycle (NDLC) method which consists of six stages: Analysis, Design, Simulation Prototyping, Implementation, Monitoring, and Management. This method is used to design, test, and implement Ansible-based server automation on Ubuntu Server systems. Data is collected through observation, pre- and post-automation performance testing, and system log analysis. Research instruments include server hardware, software such as Ansible, Docker, Kubernetes, as well as monitoring tools Prometheus and Grafana. Data analysis is carried out quantitatively by comparing deployment time, resource efficiency, network latency, and system security. It is hoped that this research can improve the efficiency, security, and scalability of network infrastructure through NDLC-based automation.

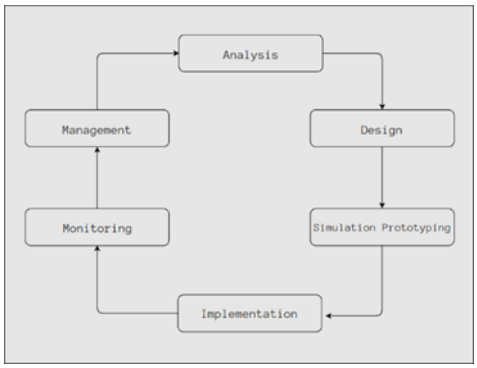

Figure 1. NDLC Model
Source: Research simulation results, (2024)

RESULTS AND DISCUSSION

A method used to describe or design a device with another device in order to communicate properly or can also be a broad overview of the data network process can be connected. Each table and graph needs to be accompanied by an explanation or analysis.

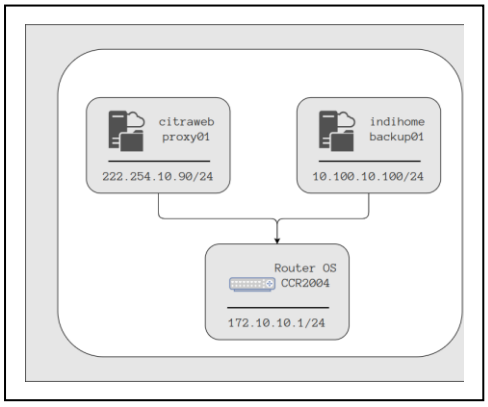

Figure 2. Topology
Source: Simulation results at PNET-Lab, (2024)

The topology of the computer network on the PNET-Lab server uses 2 providers, where each provider is interconnected with the OS router where ethernet 1 is ISP01 and ethernet 2 is ISP02. Based on the topology used, there are weaknesses in the network systems on the server, which can affect network management. This vulnerability occurs when one of the connections to the router experiences an interruption, such as being disconnected or unresponsive, causing the server to lose access to the internet network. To overcome this, good network management is required through proper routing configuration. This configuration aims to ensure that data packets can be routed from one network to another, forming a specific path or route that is effective within the routing table, while improving network reliability and efficiency. with IP address allocation 192.168.50.1-192.168.50.100, for local device allocation, main link via indihome, backup via citranet 192.168.50.101-192.168.50.254 for server allocation, main link via citranet, IP address, DHCP Server feature can be activated to provide IP automatically. To direct backups via indihome, a Failover clustering topology was created on this network using a mechanism that aims to maintain smooth operations despite a disruption to one of the network components. This process works by automatically redirecting traffic or services to the backup path or backup device, so that the system can continue to function without significant hindrance. As an illustration, in the above topology, IndiHome customers who use MikroTik devices can rely on failover to redirect the connection to a backup line, such as a cellular network or other internet provider, if the main connection of IndiHome is interrupted. MikroTik has built-in features such as Netwatch and Route Distance settings that make it easy to monitor the main route and automatically switch to an alternate route. Failover routing mechanisms are essential to ensure high network availability and reduce the risk of downtime.

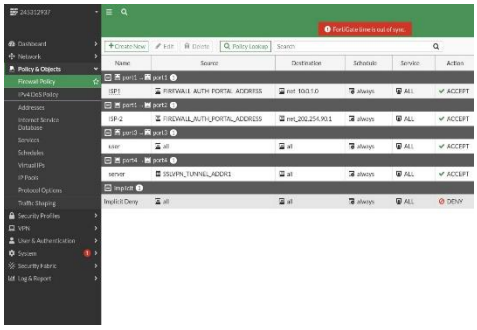

Figure 3. FortiGate
Source: Research implementation results, (2024)

FortiGate configuration involves setting up addressing and policy settings to connect the internal network to the internet securely and efficiently. The addressing mode must be adapted to the service provided by the ISP use manual mode if the ISP provides a fixed IP address, and then enter the IP and subnet mask manually. Select DHCP if the ISP automatically assigns an IP through their device. The LAN interface is set up with the LAN role, where the private IP is assigned manually. If the internal device needs traffic to the internet, it is necessary to create a default route through Network - Static Routes. This route sets the destination to 0.0.0.0/0.0.0.0, uses the ISP-provided IP gateway, and redirects through the WAN interface. Traffic policies are created through the IPv4 Policy & Objects. This policy connects LANs and WANs with configurations such as ISP1 Ingress Interface and NAT Enable LAN Egress Interface to ensure outbound traffic uses LAN interface addresses. This configuration ensures that FortiGate can efficiently manage network traffic and provide secure internet connectivity for internal devices.

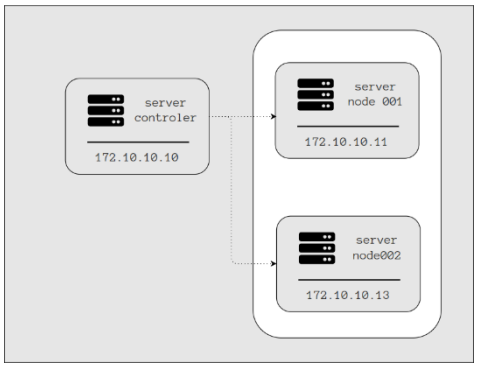

Figure 4. Repo-
Source: Research implementation results, (2024)

This stage of server installation using the GitLab repository involves managing the server infrastructure using the Server Controller as the control center for the Node001 and Node002 Servers. This process includes several key steps of Initial Connection and Validation The server controller is tasked with ensuring that the SSH connection with Node001 and Node002 runs smoothly. These checks ensure seamless authentication, allowing complete control over the nodes. The Clone Repository Repository LAMP is downloaded from GitLab to provide a playbook of LAMP installation (Linux, Apache, MySQL, PHP). Furthermore, the Docker application repository is also downloaded for container installation preparation. Deploy LAMP and Docker, the LAMP Playbook from the GitLab repository is deployed to Server Node001 to install the web application stack. Docker is installed on Node001 using the Ansible method for efficiency. After that, the Docker application is deployed to the Node001 Server using Minikube. Server Controller Monitoring (172.10.10.10): Monitored using the PING service (ICMP) to ensure the server is active and reachable. Node001 server (172.10.10.11) Monitored with PING service (ICMP) for network connectivity and HTTP (Port 80) to ensure web applications such as LAMP run smoothly. Node002 server (172.10.10.13): Monitored using the PING service (ICMP) to check the status of the server. This process ensures that the infrastructure runs stable.

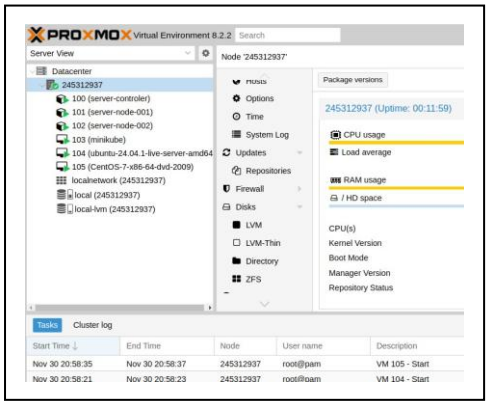

Figure 5. Proxmox Virtual Environment
Source: Research implementation results, (2024)

Proxmox is an open source software that serves as a virtualization platform for running Virtual Appliances and Virtual Machines. Proxmox VE is a special distribution designed specifically for this purpose. as a system virtualization host machine and contains 2 virtualization technologies, namely Full Virtualization (KVM) and Container Virtualization (OpenVZ). for Access to Proxmox VE via port 8006 via Secure Shell (SSH) Further the merging of three server nodes merged into one cluster can be seen in figure 3.3. Proxmox Virtual Environment The purpose of the three-server clustering process of Proxmox is to be able to use the anisible feature, which is a feature that allows automating the installation and configuration of virtual machine server set-up from one node to another. An annihilable virtual machine is a controller server that is used as a file sharing repositoty server. The server controller virtual machine serves as a host to manage migrations towards node001 and node002 servers. Ansible systems are used to access virtual machine repositories from one node to another without disrupting the running service. The use of these cluster features is designed to makes it easy for server administrators to manage multiple Proxmox nodes through a single, centralized web interface and enables efficient monitoring of virtual machines.

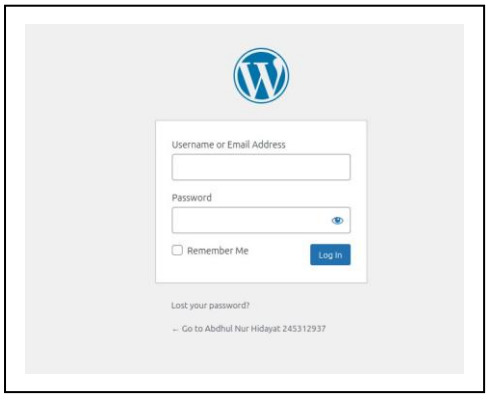

Figure 6. WordPress
Source: Results of research implementation and deployment, (2024)

Installation process involves the steps of containerization of WordPress and MySQL databases, configuration of Kubernetes services, and proxy settings using NGINX to ensure applications can be accessed stably and securely. Configure NGINX as a Reverse Proxy Create an NGINX configuration file that directs traffic to the WordPress service. Deploy NGINX as a

container or as part of a Kubernetes cluster. Expose Applications with Load Balancers Use the LoadBalancer service type on Kubernetes to allow external access to WordPress applications. Install Verification Open the browser and access WordPress via the IP or URL provided by Load Balancer. Follow the WordPress configuration wizard to complete the setup. With this architecture, you get a lightweight, flexible, and scalable solution to run WordPress efficiently in an on-premises Kubernetes environment.

Table 1. Virtualized Server

Metric	Virtualization Performance Benchmark		
	Bare Server	Virtualized (KVM)	Improv
Boot Time	25s	18s	+28%
CPU Usage	80%	70%	-12.5%
Disk I/O	500 MB/s	450 MB/s	-10%

Source: Research test results, (2024)

A performance comparison between bare metal servers and servers virtualized using KVM is based on three key metrics: boot time, CPU utilization, and disk I/O performance. The results showed that servers virtualized with KVM had faster boot times, which was 28% faster compared to bare metal servers. This signifies that virtualization can speed up the system start-up process, which is especially useful in scenarios that require shorter operational times. In addition, the CPU usage in virtualized servers is also more efficient, with a 12.5% decrease in CPU usage, which means that CPU resources are more optimally utilized in virtualized environments.

However, despite the efficiency improvements on some metrics, disk I/O performance on virtualized servers has decreased by about 10% compared to bare metal servers. This decline may be due to the additional overhead introduced by the virtualization layer, although it is still within acceptable limits for most applications. Overall, virtualization using KVM provides advantages in terms of boot speed and reduced CPU usage, although further optimization of disk I/O performance may be required to achieve more optimal results in some use cases.

CONCLUSION

To achieve the expected results, the researcher has taken various steps that include analysis of available data, research budget planning, configuration of the research environment, as well as a series of other preparations until the research can be completed. The results show that: (1) Based on the tests conducted, Ansible has proven to be effective in a pre-designed research environment, which is demonstrated through its ability to automate and configure faster and more efficiently than manual methods; (2) In terms of resource usage, this approach does not require large consumption because Ansible only utilizes small scripts without the need for special agents, where bandwidth requirements depend only on the commands executed; and (3) Virtualization with KVM provides advantages in boot speed and CPU usage efficiency, although there is a slight decrease in disk I/O performance that still needs further optimization. Based on these findings, it is recommended for further research to conduct more in-depth optimization of the virtualization field to improve disk I/O performance, explore the integration of Ansible with other orchestration tools such as Terraform for more comprehensive infrastructure management, and conduct trials in large-scale production environments to test the resiliency and scalability of the proposed solution.

REFERENCE

Arunawati, A. P. (2020). Apache Web Server Optimization Using Varnish Web Cache and Reverse Proxy Nginx. Thesis, Department of Computer Science, Faculty of Mathematics and Natural Sciences, Semarang State University.

Egbuna, P. O. (2022). Security challenges and solutions in Kubernetes container orchestration. Journal of Science & Technology, 3(3), 66-90. https://thesciencebrigade.com/jst/article/view/233

Hariyadi, I. P., & Marzuki, K. (2020). Implementation of configuration management virtual private server using Ansible. Journal of Information Systems Technology, 14(5), 67–80.

Hasbi, M., Nurwa, A. R. A., Priambodo, D. F., Putra, W. R. A., Nusantara, S. S., & Negara, P. S. S. (2022). Infrastructure as Code for security automation and network infrastructure monitoring. Teknik Informatika Dan Rekayasa Komputer, 22(1), 203-217. https://doi.org/10.30812/matrik.v22i1.2471

Islami, M. F., Musa, P., & Lamsani, M. (2020). Implementation of network automation using Ansible to configure routing protocol in Cisco and Mikrotik router with Raspberry Pi. Journal of Network Automation, 25(2), 145–160.

Khumaidi, A. (2021). Implementation of DevOps method for automation of server management using Ansible. Journal of Software Engineering and Automation, 12(3), 221–234.

Kristianto, L. Y. (2023). Designing a virtualization system-based server optimization using VMware software. Final project, Informatics Study Program, Faculty of Industrial Technology, Atma Jaya University, Yogyakarta.

Pratama, M. A. A., & Hariyadi, I. P. (2021). Automation of Linux container management and supervision (LCX) on Proxmox VE using Ansible. Journal of Computer Engineering, 31(4), 125–140.

Putri, Y. M., Aditya, M., & Santosa, F. (2021). Ansible for automated deployment and containerization management. Journal of Automation and Systems Engineering, 39(6), 322–335.

Putro, R. A., & Supono, S. (2022). Container-based virtualization and Kubernetes: Challenges and opportunities for organizations. Journal of Cloud Computing and Virtualization, 34(2), 98–110.

Ramdhani, A. I., Subekti, Z. M., Putro, E. M., Jaya, I., & Ramadhan, A. (2023). Automate web service configuration on Ubuntu server using Python-based Ansible. Journal of Devices, 13(1), 88–99. https://doi.org/10

Rina, L., & Ridha, A. (2021). Strengthening network security with FortiGate firewalls in enterprise environments. International Journal of Network Security, 41(2), 120–133.

Sethuraman, S. C., Gayathri, N., Raman, A. R., Hazra, B., & Mohanty, S. P. (2023). On the security of containers: Threat modeling, attack analysis, and mitigation strategies. Computers & Security, 130, Article 103263. https://doi.org/10.1016/j.cose.2023.103263

Sharma, A., Kumar, R., & Patel, S. (2019). Container security in the cloud: Hardening orchestration platforms against emerging threats. World Journal of Advanced Research and Reviews, 4(1), 64-74. https://doi.org/10.30574/wjarr.2019.4.1.0074

Singh, P., Johnson, M., & Chen, L. (2023). Container-based virtualization for real-time industrial systems—A systematic review. ACM Computing Surveys, 56(2), 1-35.

Sun, J. (2022). Network management challenges and solutions in modern business operations. Journal of Network and System Administration, 45(3), 215–230.

Thompson, K., Anderson, J., & Williams, R. (2023). Systematic analysis of Infrastructure as Code technologies. IEEE Transactions on Network and Service Management, 20(4), 1245-1260. https://www.researchgate.net/publication/376476103_Systematic_Analysis_of_Infrastructure_as_Code_Technologies

Wijaya, H., Abdurrohim, L., Tugiyono, J., & Rumandan, R. J. (2023). Implementation of the load balancing method for optimizing server performance on the internet network. Journal of Information Systems Optimization, 22(1), 50–65.

Wicaksono, A., & Widiasari, D. (2022). Implementing FortiGate firewalls for enhanced network security. International Journal of Cybersecurity and Information Systems, 29(4), 198–211.

Yalestia, N. M. A., Chandrawaty, & Hariyadi, P. (2021). Ansible playbook implementation to automate VTP-based VLAN configuration management and DHCP services. Journal of Information Technology and Computer Systems, 19(1), 100–115.