Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer
Vol. 22, No. 1, November 2022, pp. 201∼214
ISSN: 2476-9843, accredited by Kemenristekdikti, Decree No: 200/M/KPT/2020
DOI: 10.30812/matrik.v22i1.2471

Infrastructure as Code for Security Automation and Network Infrastructure Monitoring

Muhammad Hasbi[1], Agus Reza Aristiadi Nurwa[2], Dimas Febriyan Priambodo[3], Wahyu Riski Aulia Putra[4]
[1]STMIK Sinar Nusantara, Surakarta, Indonesia
[2,3,4]Politeknik Siber dan Sandi Negara, Bogor, Indonesia

Article Info

Article history:

Received July 27, 2022
Revised October 12, 2022
Accepted November 27, 2022

Keywords:

Ansible-playbook
Infrastructure as code
Network Monitoring
Security automation
System administration

ABSTRACT

The Corona Virus (COVID-19) pandemic that has spread throughout the world has created a new work culture, namely working remotely by utilizing existing technology. This has the effect of increasing crime and cyber attacks as more and more devices are connected to the internet for work. Therefore, the priority on security and monitoring of network infrastructure should be increased. The security and monitoring of this infrastructure requires an administrator in its management and configuration. One administrator can manage multiple infrastructures, making the task more difficult and time-consuming. This research implements infrastructure as code for security automation and network infrastructure monitoring including IDS, honeypot, and SIEM. Automation is done using ansible tools to create virtual machines to security configuration and monitoring of network infrastructure automatically. The results obtained are automation processes and blackbox testing is carried out and validation is carried out using a User Acceptance Test to the computer apparatus of the IT Poltek SSN Unit to prove the ease of the automation carried out. Based on the results of the UAT, a score of 154 was obtained in the Agree area with an acceptance rate of 81.05% for the implementation of infrastructure as code for the automation carried out.

Corresponding Author:

Dimas Febriyan Priambodo, 081226400058
Department of Cyber Security Engineering,
National Cyber and Crypto Polytechnic. Indonesia
Email: dimas.febriyan@poltekssn.ac.id

How to Cite: M. Hasbi, A. R. Nurwa, D. Priambodo, and W. R. Putra, Infrastructure as Code for Security Automation and Network Infrastructure Monitoring, MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer, vol. 22, no. 1, pp. 203-217, Nov. 2022.

1. INTRODUCTION

The Corona virus (COVID-19) pandemic that has spread throughout the world has resulted in a new culture for the traditional work environment, namely a work from home culture by utilizing existing technology [1]. WFH work culture has an impact on increasing cyber attacks [2]. Cyber attacks that occurred in Indonesia in 2020 amounted to 316 million attacks [3] or 3 times more than in 2019 before this remote work culture increased. Cyber attacks that occur also vary from malware attacks, SSH attacks, and Web-based attacks. This makes the priority of security and monitoring in the network infrastructure must be increased. Network infrastructure security is the process of protecting devices or infrastructure connected to the network by installing precautions against modification, and deletion of unauthorized resources or data. This infrastructure security can be in the form of intrusion detection systems, intrusion prevention system and honeypots so that with the existence of network infrastructure security can detect or prevent cyber attacks. Meanwhile, network infrastructure monitoring is a process that monitors the availability, uptime, operation, and performance of network infrastructure. This process includes tracking and analyzing network infrastructure components such as routers, switches, access points, and security network infrastructure such as honeypots, intrusion detection systems, and firewalls.

Infrastructure design is a phase of software development as the number and type of virtual machines required. Implementing network infrastructure security and monitoring has the advantage of being able to detect or prevent cyber attacks . In its application, network infrastructure security can be in the form of servers or virtual machines that function as honeypots or intrusion detection systems [4]. Meanwhile, network infrastructure monitoring can be in the form of Security Information and Event Management (SIEM). Honeypot is a system or computer designed to be the target of hacker attacks [5]. Honeypot aims to find out the attack techniques used so that it becomes a lesson for the future and can also protect the real server [6]. The information obtained from the honeypot can be used as a basis for establishing or updating the network security perimeter so that the same attack will not affect the original system. Meanwhile, intrusion detection (IDS) is a system used to detect legitimate intrusions into computer systems and networks. IDS can also be defined as a security application for computers and networks that is able to collect and analyze data by preventing incoming or outgoing network traffic and detecting suspicious activity using certain rules, and will alert users if they find malicious activity. Meanwhile, Security Information and Event Management (SIEM) is a monitoring system that functions to detect attacks and responses to a security system through log analysis of various events originating from various data sources that are connected in real time [7]. SIEM will be connected to various devices that can provide information about logs and attack events that occur to these devices. Devices such as honeypots and intrusion detection systems can be linked to SIEM. If the honeypot is attacked or there is malicious traffic detected by the intrusion detection system, logs and alerts will be generated which will be visualized by SIEM. So that every attack or anomaly can be immediately addressed by reviewing the existing security system to avoid further damage [7].

The security and monitoring of this infrastructure requires an administrator in its management and configuration. One administrator can manage many infrastructures, making the task more difficult and time-consuming [8]. Management and configuration of network infrastructure security takes a long time because of the many commands that must be executed [9], repetitive configurations, and many servers that must be configured at the same time [8]. In addition, the management and configuration of network infrastructure security also requires a deep understanding so that it can work as needed.

Based on these problems, there is one solution, namely the implementation of infrastructure as code to simplify and make the work of network infrastructure administrators faster and more efficient [10]. Infrastructure as code (IaC) is a technique that - defines computer network infrastructure in the form of source code-based automation, aiming to provide an infrastructure for a software development environment, configuration, management, and network infrastructure by taking into account security aspects [11]. IaC emphasizes the automation of management and provision of infrastructure through a file or code that can be read by a computer so as to reduce manual configuration on computers and networks [12]. Implementing IaC in IT organizations increases the deployment frequency by 21 times and makes it possible to manage 15,000 servers and 2000 TB of data every day [13].

As mention above Matej Artac, et al [12] Introducing infrastructure as code, this study explains that infrastructure design is one of the phases/stages in the ”software lifecycle” where at this stage will define and configure the infrastructure needed for applications to be run such as many virtual required machine, virtual machine type, required dependencies, and so on. Infrastructure as Code (IaC) is the process of designing infrastructure so that the entire series can be done automatically through the scripts used. In line of automation Ali Khumaidi [8] conducted research by implementing one of the infrastructure tools as code, namely ansible for the management of several servers simultaneously and automatically. create a file and directory simultaneously and automatically. Bongga Arifwidodo, et al touch a little by implementing infrastructure as code for web server automation in a cloud environment using terraform tools. The result obtained is that the process of making web server infrastructure using terraform takes more than 12 minutes 19 seconds than manually. I Putu Hariyadi [16] also conducted research on the automation of Virtual Private Server configuration (VPS) on proxmox using the Infrastructure as Code with ansible. This research creates an ansible-playbook that can automate several tasks such as creating or running containers, changing user passwords, deleting pools, and group containers. This

study creates 93 VPS automatically and simultaneously and the results obtained are that making 1 VPS only takes 26.25 seconds or the total to create 93 VPS is 40.3 minutes so that the use of ansible is very helpful for multiple and repetitive server configurations. One more research about automation from Amous O Olagunju, et al [14] conduct research by making honeynet and honeypot automatically using infrastructure as code tools, namely puppet. The results obtained from the research are honeypots that are created automatically can work well by detecting 498,471 attacks for 10 days and obtaining some information such as the attacker's origin, top 10 passwords, and top 10 usernames used by the attacker.

Based on this background, this research will compile all research above to implement infrastructure as code for network infrastructure include virtual machine create and auto configuration, security automation in the form multi IDS using Suricata and Zeek has not been implemented before, also using automatic honeypot as research Amous O Olagunju, et al [14] by adding dual honeypot Cowrie and Dionaea using ansible not puppe. In Final presentation using auto configuration of SIEM using Elastic search, Logstash and Kibana (ELK). The final results of the research were then validated with a User Acceptance Test (UAT) to the Information Technology unit of the SSN Poltek to find out whether the system built made it easier to manage and configure security and monitor network infrastructure.

Suricata was chosen because it can handle more packets with its multi-thread feature so that it can detect intrusions in real time [7] while Zeek can record network activities including HTTP sessions, DNS requests, malware attacks, SSH brute-force attacks, and validation. certificate in log form. Honeypot Cowrie was chosen because it can categorize threats that occur on SSH-based systems (Secure Shell) or simulate SSH services. Cowrie can also record and generate command execution logs from attackers on the system, so that it can monitor the behavior/patterns of attacks by attackers [8]. Whereas Dionaea can trap exploiting malware through vulnerabilities that are intentionally created and designed to attack, the purpose of which is to obtain a copy of the malware that the attacker is using. The automation created is expected to be used to simplify the management and configuration of security and monitoring of network infrastructure using infrastructure as code tools.

2. RESEARCH METHOD

Distributed system consisting of several computers to communicate with each other using communication media [12]. A network is built with hardware and software configurations to work properly. Connected devices can be thought of as a network there must be rules or communication protocols that each connected device follows. Network infrastructure refers to all network resources that enable network or internet connectivity, management, business operations, and communications. Network infrastructure consists of hardware and software, systems and devices, and enables computing and communication between users, services, applications, and processes. While security is an ongoing process that aims to protect objects from unauthorized access. This object can be a person, organization, computer system, and file [12]. Network security refers to the protection of hardware, software, and data on network systems so that they are not changed, leaked, and damaged against an incident or other cause of crime [15]. Network infrastructure security is the process of protecting devices or infrastructure connected to the network by installing precautions against modification, and deletion of unauthorized resources, in this research implementeted dual IDS dual honeypot and reporting using SIEM.

Infrastructure as code (IaC) is a technique that defines computer network infrastructure in the form of source code-based automation, with the aim of providing an environment for software development, configuration, management, and network infrastructure with regard to security aspects [11]. IaC emphasizes the automation of management and provision of infrastructure through a file or code that can be read by a computer so as to reduce manual configuration on computers and networks. Using IaC can simplify the process of preparing a system that is needed, especially if a large number of systems are needed. The use of IaC also makes every system that is made consistent and the same because it uses a script that runs the same thing. Infrastructure as code workflow is explained in Figure 1 where when creating infrastructure using the Infrastructure as Code concept, there are workflows that can be used to facilitate the process or the running of an automated job. Workflow can be started from the user writing code and generating infrastructure code. Like the process of making an application in the form of code, the results of the code can use version control to help manage and change the code that is made. After the code is considered complete and can be used, the code can be run to automate the desired server infrastructure according to what is defined in the code that was created.

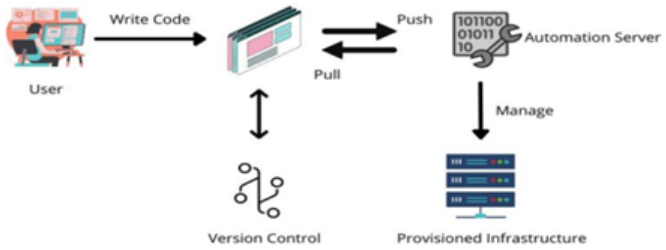


Figure 1. Infrastructure as code Workflow

The research method that will be used to implement infrastructure as code is the System Development Life Cycle. System Development Life Cycle (SDLC) is a method and guide used to determine how to manufacture and develop to produce a quality information system in accordance with the wishes of users [16]. This research is also part of the creation of an information system but in another form, namely automatic code generation to facilitate deployment so that the SDLC is deemed suitable. The SDLC method consists of four stages as Planning, Analysis, Design, Implementation, and maintenance. The stages in the SDLC are described in Figure 2.

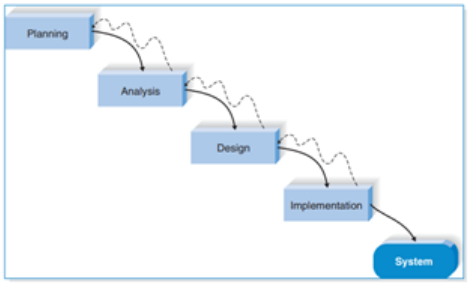


Figure 2. Effects of selecting different switching under dynamic condition

2.1. Planning

The Planning stage is a process that understands the reason the system was created and determines how the system is made and knows the problem and what solutions can solve the problem [16]. At this stage, an overview of network infrastructure security automation for security and network infrastructure monitoring using ansible. Automation is carried out to simplify the process of creating and deploying infrastructure such as honeypots and intrusion detection systems and connecting them to Security Information and Event Management (SIEM) so that they can be monitored if there is suspicious traffic.

Honeypot is sysem or computer designed in such a way that it can attract attention and be hacked by attackers [13]. When the honeypot is attacked, the attacker's methods and tools will be known so that it can be used as learning material for the honeypot owner. Honeypot will emulate a system that has services and vulnerabilities that have the potential to be attacked, honeypot will receive attacks on the system and accumulate attacks and record them to analyze the attack methods carried out to assist in optimizing the system in the future. Although honeypots are created to protect real systems and are made to look as similar as possible in order to attract the attention of attackers, honeypots contain no valuable or only false information about genuine systems. Honeypot has a level of interaction that is used to measure the amount of activity carried out by the honeypot with the attacker [15]. The higher the activity allowed for the attacker, the higher the honeypot interaction [13]. Some examples of Honeypots that are widely used are Cowrie and Dionae.

Cowrie is a Python-based high-interaction honeypot developed by Michel Oosterhof in 2015. Cowrie can be used to detect and log shell interactions and brute force attacks such as SSH and Telnet which are commonly used by attackers [9]. Cowrie uses ports 22 and 23. Cowrie allows an attacker to initiate a connection and login to the system and trick the attacker as if they were actually entering the system through a brute force login attack. If the login is successful, the attacker will be redirected to a decoy operating system that can be used to interact with the attacker [15]. When interacting with an attacker, Cowrie captures all events to record and analyze.

Dionaea is a low-interaction honeypot developed by the Google Summer of Code (GSoC) 2009 from the Honeynet Project using the Python programming language. The Dionaea honeypot will collect attack information in a log file that can be used to analyze further attacks to help understand the attacks captured by the honeypot [17]. Dionaea traps the exploiting malware through a vulnerability it is designed to attack, the purpose of which is to obtain a copy of the malware. Dionaea will work by opening multiple ports which are vulnerabilities that can be exploited by attackers, the malware that Dionaea catches will be stored in a folder.

Interusion Detection System (IDS) is system used to detect unauthorized intrusions into computer systems and networks [13]. IDS can also be defined as a security application for computers and networks that is able to collect and analyze data by preventing incoming and outgoing network traffic and detecting malicious activity using certain rules, and will provide warnings to users if they find malicious activity. IDS is an integral part of network security because IDS implements an in-depth defense strategy on the network by identifying and mitigating attacks accompanied by logging data for future analysis. Host-Based IDS (HIDS) is a type of IDS that is configured inside a host machine on a network and performs scans only on the host to detect malicious activity on the host. Network-Based IDS (NIDS) is an IDS that is installed on a physical network device and scans all incoming and outgoing packets on the network to detect any unauthorized access attempts on the network [15]. Hybrid IDS is a combination of HIDS and NIDS that is configured to monitor all data packets across a network. Some examples of intrusion detection systems that are widely used are Suricata and Zeek.

Suricata is software developed by the Open Information Security Foundation (OISF) in 2010. Suricata has multitasking capabilities so it can work on more packages or in other words Suricata has a multithreading feature that allows more than one processing [9]. Figure 3 shows packet processing on Suricata. Suricata has real-time intrusion detection (IDS), inline intrusion prevention (IPS) and Network Security Monitoring (NSM) capabilities. Suricata verifies network traffic using rules and signatures. Suricata has standard input and output formats such as YAML and JSON integration that can be used with other tools such as SIEM, Splunk or ELK stack [9].

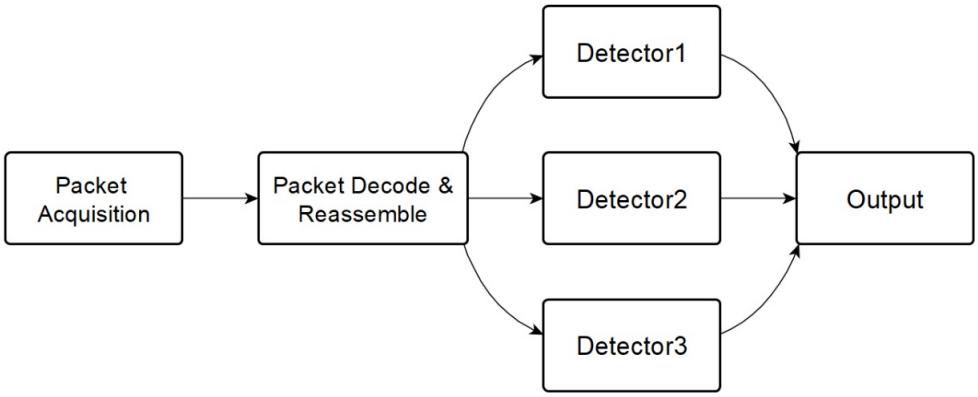


Figure 3. Architecture of Packet Processing Suricata

Zeek was first developed by Lawrence Berkeley National Laboratory (LBNL) researcher Vern Paxson in 1995. Zeek is an open source IDS and supports a wide range of applications to perform network packet analysis and help with troubleshooting. Zeek comes packed with scanning and detection features, including extracting files from HTTP sessions, detecting malware by connecting to external registrars. Zeek (Bro) has his own programming language called ”Bro Language” which is used to write scripting rules for Bro. The log files generated or generated by Zeek are divided into sections based on the results from the protocol analyzer and file analyzer.

SIEM as we knows is Monitoring software that functions to record, monitor, warn, anticipate, connect and display the security of events and information on network systems. This SIEM technology has a large and centralized data collection scope and is able to correlate and analyze events from various sources and determine whether the incident is an attack or not. SIEM has two layers, namely a layer for log management functions and a layer for security analytics [20]. Activities at these two layers are distributed among the Security Information Management (SIM) and Security Event Management (SEM) components of SIEM. SEM performs real-time

monitoring, event correlation, and incident response, while SIM collects, stores, analyzes, and reports log data. One example of a widely used SIEM is the ELK stack and Wazuh [21].

This system will be implemented on a PoltekSSN server connected to the PoltekSSN bridge network and on a router to simulate a public internet network and a private router network. The honeypot used is dioanea and Cowrie, while the Intrusion Detection System used is Snort and Zeek and the Security Information and Event Management used is the ELK stack with Wazuh [21] as shown in Figure 4. A, B, C and D is for Honeypot, Virtual Machine, SIEM and IDS Configuration Automation.

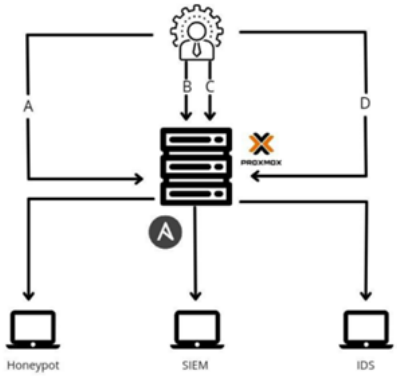


Figure 4. Planning

2.2. Analysis

The analysis stage is a stage that aims to determine the usefulness of the system created, who will use the system, and where and when the system is used [16]. The purpose of the requirements analysis is to explain all the requirements contained in the development of security automation and network infrastructure monitoring using Ansible. Ansible is Information technology automation tools used for application configuration management and agentless orchestration [18]. Ansible can configure systems, run software, and configure advanced information technology functions such as continuous deployment or rollback updates [18]. Ansible can use ad-hoc commands to send not too many commands to the target but if there are a lot of commands it can use ansible-playbook which is a file in yaml format. Ansible will run the appropriate commands in the playbook and will provide information on how many changes were made. The result of this stage is a detailed description of the development of security automation and network infrastructure monitoring using Ansible which will be used in the design stage. as seen in Table 1, Functional requirements are requirements that must exist and are related to the system and contain the required information and Non-functional requirements are requirements outside of the automation system requirements but have a supporting role for working automation system services.

Table 1. Functional and Non-Functional Needs

Fungsional Needs	Non Fungsional Needs
can create virtual machines automatically using the Ubuntu operating system	built using ansible
can configure intrusion detection system form Suricata and Zeek automatically	run using ansible-playbook
can configure honeypots in the form of Cowrie and Dionaea automatically	
can automatically configure security information and event management using ELK Stack and Wazuh	

2.3. Design

The Design stage is the stage of determining how the system will operate in terms of hardware, software, and user interfaces, formulas, programs, databases, and specific files that will be needed [16]. At this stage, an automation design is made for security and monitoring of network infrastructure based on the analysis that has been done. The design model uses flow diagrams to be able to visualize the processes that occur in honeypot automation, intrusion detection, and Security Information and Event Management (SIEM) using Ansible.

2.4. Implementation

The Implementation stage is the last stage in the SDLC method, consisting of three steps, namely system development, system installation, and making a support plan for the system created [16]. At this stage, the development of an automation system is carried out, namely security and monitoring of network infrastructure based on the design that has been made previously. After the system is built, it will be tested on the system. User Acceptance Test (UAT) is one of the stages of software or system development to test and ensure whether its functions and tasks are in accordance with user requirements or needs [19]. The main focus of UAT is to ensure that users feel comfortable when using the product and can solve user problems. One way to get the UAT value is to collect data using a questionnaire.

This questionnaire is intended to collect user responses related to the system and as confirmation to users whether the system built is in accordance with the needs, in this case the IT Unit in National Cyber and Crypto Polytechnic. This test distributes a questionnaire containing questions about whether the functionality of the system has made it easier and in accordance with the needs of the system administrator using Likert scale. Likert scale which has user answers on a scale of 1 to 5 with gradations from Strongly Agree (SS), Agree (S), Doubtful (RG), Disagree (TS) and Strongly Disagree (STS. The questionnaire provides answers to questions according to the scale above. Answers given by respondents will be calculated using the formula 1. the results of these calculations are added up and recalculated in order determine the level of acceptance of the software or system on a regular basis whole by formula (2).

$$w = \Sigma a \times b \tag{1}$$

w = total score of the respondent's choice
a = total number of respondents
b = score from the Likert number

$$\frac{\Sigma w}{\text{x xy xz}} \times 100\% \tag{2}$$

w = total score of the respondent's choice
x : total statements
y = total respondents
z = highest score

3. RESULT AND ANALYSIS

3.1. Design and Implementation

The implementation environment contains an explanation of the devices used in implementing the previously designed automation system. The specifications of control node and server can be seen in Table (2). Server/Hardware uses a virtual machine with Proxmox VE as the Hypervisor, while the control node computer uses VirtualBox as the Hypervisor. The purpose of this automation is to create a virtual machine consisting of virtual machines for intrusion detection systems, honeypots, and SIEM. Virtual Machine in control node device has a role as a control node for the automated server. This virtual machine will make a configuration so that it can connect to the proxmox VE server and can make a passwordless SSH connection. The automation script is run on this virtual machine and the script is executed by the proxmox server and generates several virtual machines such as the research plan.

Table 2. Hardware and Software Spesification

Control Node	Server
Intel Core i7-8550U	Intel Xeon E3-1200
DDR3 16GB	DDR3 32 GB
Hardisk 512GB	Hardisk 1 TB
Ubuntu OS and Ansible v 2.9.6	Debian OS, PVE v 7.1-4, Promoxer module v 6.4

Implementing automated virtual machine creation on the Proxmox hypervisor Automated virtual machine creation creates a cloud-init and the script asks for input for the vmid value name, ID, Socket/cores, Memory, password, user, IP address, disk. The next automation process is cloning the previously created cloud-init virtual machine and entering the required virtual machine information. If the automation process goes well, a new virtual machine with the information previously entered in the proxmox Virtual Environment Figure (5). The variables are used in the virtual machine cloning process where the virtual machine that is

created will be in accordance with the variables entered before the virtual machine turns on without having to do manual configuration such as specifying a user, password, and IP address so that the automation process can be carried out without having to do manual configuration. to the newly created virtual machine. System automation is used using an Ansible script that requires an SSH connection so that at this stage it will also register the control node's SSH key so that the control node can make SSH connections without having to enter a password.

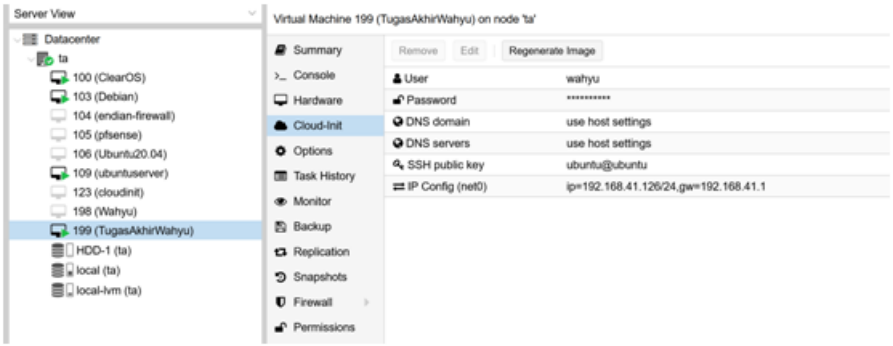


Figure 5. Cloud-Init Automation Process

3.2. Intrusion Detection System Configuration Automation

The control node automates the configuration of the virtual machine that can't be used as an IDS until it can and just use it. There are two IDSs used, namely Suricata and Zeek. Suricata Automation will add a repository from Suricata so that it can be installed using the apt package manager. After Suricata is installed, users can add rules that they want to use, in this study adding rules where Suricata will send alerts if there is an ICMP protocol in the network. Zeek Automation will first download the dependencies needed to perform a Zeek installation such as Cmake, Libssl-dev, Make, Python-dev, Gcc, Swig, G++, Zlib1g-dev, Flex, Libpcap-dev. After the dependencies have been successfully downloaded and installed on the system, the next step to take is to add the repository from Zeek so that you can install it using the apt package manager. After successfully installing Zeek, you need to change the permissions in the Zeek installed directory so that normal users (non-root) cannot use the Zeekctl command. After everything has been successfully done the next thing to do is run Zeek using the Zeekctl deploy command. If the automation process goes well, there will be a new directory according to the IDS used and services that can be turned on Suricata and Zeek Figure 6.

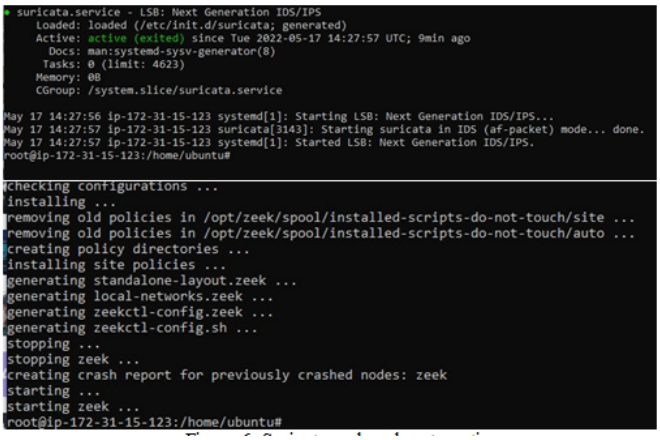


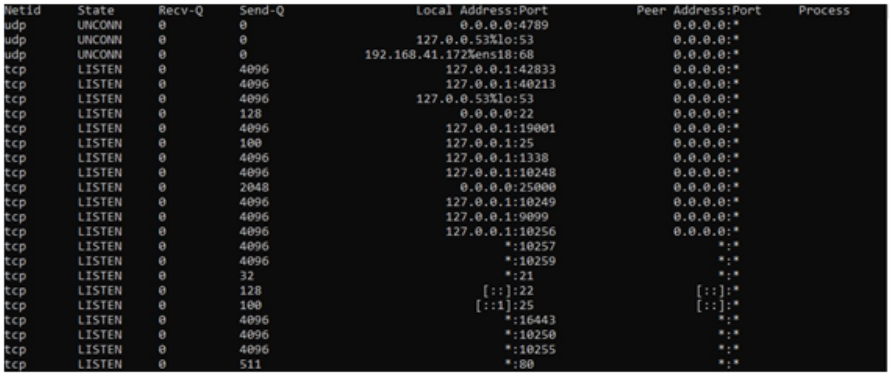
Figure 6. Suricata and zeek automation

3.3. Honeypot Configuration Automation

The control node automates the configuration of the virtual machine that can't be used as a honeypot until it can and just use it. There are two honeypots used, namely Cowrie and Dionaea. Cowrie Automation starts with downloading the dependencies needed to run the Cowrie honeypot such as git, virtualenv, libssl-dev, libpython3-dev, pyrhon3-minimal, authbind. After the dependencies have been successfully downloaded and installed on the system, the next step is to download the git repository from Cowrie which can be downloaded via the URL http://github.com/Cowrie/Cowrie. In the Cowrie git repository there is a file with the name requirements.txt where the contents of the file are the python package that Cowrie needs so that it can run properly so the next step is to create a python virtualven and install the required packages in the newly created python virtualenv. Cowrie uses port 22 as bait for spoofing SSH connections against attackers. To make the honeypot look genuine like on a server in general, the port that must be used to access SSH connections is port 22 where to make all port 22 connections diverted to port 22, you need to configure iptables in the chain prerouting port 22 to port 2222. So that the server can still run accessed by users using an SSH connection, it is necessary to change the default SSH port used from pot 22 to any port above port 1000 in this case using port 3333. This can be done by changing the ssh configuration file in the /etc/ssh/sshd_config file by uncomment port 22 text and change it to port 3333 figure 7. The last thing to do in Cowrie automation is to run services from Cowrie using the Cowrie start command in the Cowrie/bin/ directory.

Figure 7. SSH Connection Default Port Change Configuration

Dionaea Automation will first download dependencies like libemu, libssl-dev, libemu-dev, libudns-dev, build-essential, python3, cmake, python3-dev, cython3, python3-bson, libcurl4-openssl-dev, python3-yaml, libev -dev, python-boto3, libglib2.0-dev, libtool, libloudmouth1-dev, libnl-3-dev, libnetfilter-queue-dev, libpcap-dev, fonts-liberation. After the dependencies have been successfully downloaded and installed on the system, the next step to take is to download the git repository from Dionaea which can be downloaded via the url http://github.com/dinoTools/Dionaea. After that, a new directory named build will be created where this directory will contain the compile files from Dionaea. In the build directory, we will compile the Dionaea binary file using cmake and make so that Dionaea can be installed on the system and can be used. If the binary compilation process is successful then Dionaea can be run using the Dionaea -D command. If Dionaea is running well, many vulnerable ports will be opened and used to lure attackers according to the use of the honeypot itself.If the honeypot automation process goes well, there will be several ports and services open to lure attackers to attack the honeypot Figure 8.



Netid	State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port	Process
udp	UNCONN	0	0	0.0.0.0:4789	0.0.0.0:*	
udp	UNCONN	0	0	127.0.0.53%lo:53	0.0.0.0:*	
udp	UNCONN	0	0	192.168.41.172%ens18:68	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:42833	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:40213	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.53%lo:53	0.0.0.0:*	
tcp	LISTEN	0	128	0.0.0.0:22	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:19001	0.0.0.0:*	
tcp	LISTEN	0	100	127.0.0.1:25	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:1338	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:10248	0.0.0.0:*	
tcp	LISTEN	0	2048	0.0.0.0:25000	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:10249	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:9099	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.1:10256	0.0.0.0:*	
tcp	LISTEN	0	4096	*:10257	*:*	
tcp	LISTEN	0	4096	*:10259	*:*	
tcp	LISTEN	0	32	*:21	*:*	
tcp	LISTEN	0	128	[::]:22	[::]:*	
tcp	LISTEN	0	100	[::1]:25	[::]:*	
tcp	LISTEN	0	4096	*:16443	*:*	
tcp	LISTEN	0	4096	*:10250	*:*	
tcp	LISTEN	0	4096	*:10255	*:*	
tcp	LISTEN	0	511	*:80	*:*	

Figure 8. Honeypot Automation Results

3.4. Security Information and Event Management Configuration Automation

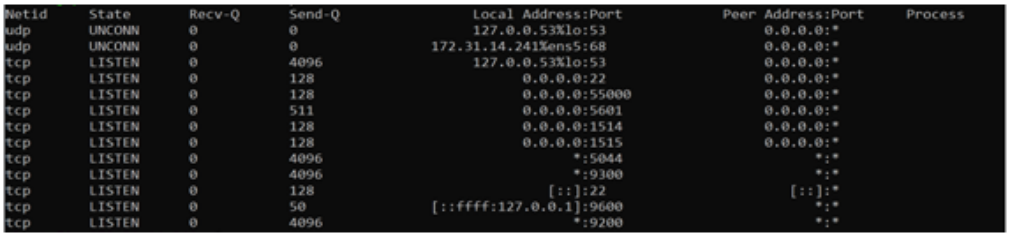
Wazuh can run if the ELK stack has been installed so the automation process that is carried out first is the installation and configuration of the ELK stack. ELK Stack Automation will first download the dependencies needed to perform the ELK stack installation. Required dependencies like Curl, Gnupg2, Apt-transport-htpp, Default-jre, Unzip, Lsb-release, Wget, Libcap2-bin, Software-properties-common. If the dependencies are downloaded successfully, the next step is to automate the installation of elasticsearch, logstash, kibana, and filebeat by downloading the deb package via the URL provided by elastic according to the version required by the user. After the deb package has been successfully downloaded, the deb package can be installed on the system so that elasticsearch, logstash, kibana, and filebeat can be used.

If all processes are running well, the elasticsearch service can be turned on and can be accessed by other services such as Logstash and kibana. Next, install the logstash by downloading the deb package via the URL provided by elastic according to the version required by the user. After the deb package has been successfully downloaded, the deb package can be installed on the system so that the logstash can be used. Logstash requires a pipeline file so that a file with the name main.conf will be created in the /etc/logstash/conf.d/main.conf directory and then filled with information in the form of the port used, and the IP address of elasticsearch that was installed before.

In Kibana, the file in /etc/kibana/kibana.yml can be changed by adding the IP address from elasticsearch along with the port used. As for Filebeat, a file will be provided on the control node named filebeat.yml which already contains the information needed in the form of activating the configuration input log along with the log path to be received, the IP address of Kibana and the elasticsearch server. This beat.yml file will be sent to the ELK stack server in the /etc/filebeat/filebeat.yml directory and delete the default configuration file.

If the ELK stack is installed and running properly, the Wazuh installation and configuration automation process can be carried out. To automate the installation and configuration of Wazuh, start by adding the repository from Wazuh so you can install Wazuh-manager using the apt package manager. Because Wazuh runs on top of Kibana, the next step is to create a new directory for Wazuh in the /usr/share/kibana/data directory and then install the plugin from Wazuh in that directory.

The next step is so that Wazuh can receive log information from the endpoint being monitored Figure 9, there is a filebeat configuration for Wazuh where in the previous beat.yml file several configurations will be added to activate the Wazuh module in filebeat. After everything goes well, when accessing the kibana dashboard, there will be a Wazuh option to access the Wazuh service. If the automation process goes well then there are several ports and services open to access SIEM such as kibana and Wazuh dashboard.



Netid	State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port	Process
udp	UNCONN	0	0	127.0.0.53%lo:53	0.0.0.0:*	
udp	UNCONN	0	0	172.31.14.241%ens5:68	0.0.0.0:*	
tcp	LISTEN	0	4096	127.0.0.53%lo:53	0.0.0.0:*	
tcp	LISTEN	0	128	0.0.0.0:22	0.0.0.0:*	
tcp	LISTEN	0	128	0.0.0.0:55000	0.0.0.0:*	
tcp	LISTEN	0	511	0.0.0.0:5601	0.0.0.0:*	
tcp	LISTEN	0	128	0.0.0.0:1514	0.0.0.0:*	
tcp	LISTEN	0	128	0.0.0.0:1515	0.0.0.0:*	
tcp	LISTEN	0	4096	*:5044	*:*	
tcp	LISTEN	0	4096	*:9300	*:*	
tcp	LISTEN	0	128	[::]:22	[::]:*	
tcp	LISTEN	0	50	[::ffff:127.0.0.1]:9600	*:*	
tcp	LISTEN	0	4096	*:9200	*:*	

Figure 9. SIEM Automation Results

3.5. Intrusion Detection System and Security Information and Event Management Integration

The automation process of connecting the Intrusion Detection System and Security Information and Event Management can be done after the filebeat configuration file has been changed so that it can connect with SIEM, the next thing to do is activate the filebeat module from each IDS. The script for connecting the Intrusion Detection System and Security Information and Event Management Figure 10.

```
- name: Enable module Suricata
  command: filebeat modules enable Suricata
- name: Enable module Zeek
  command: filebeat modules enable Zeek
- name: Setup filebeat
  command: filebeat setup
- name: start service filebeat
  ansible.builtin.service:
    name: filebeat
    state: started
```

Figure 10. Script connecting IDS and SIEM

3.6. Honeypot and Wazuh Integration

The automation process connecting the honeypot and Wazuh will add the repository from Wazuh so that you can install Wazuh-agent using the apt package manager. After the Wazuh agent is installed on the system, it will then run the command /var/ossec/bin/agen-auth -m MANAGERIP, MANAGER-IP can be filled with the IP address of the Wazuh server. This command functions to register the Wazuh manager used by the honeypot. The next step that the script will take is changing the configuration file of the Wazuh-agent in the file /var/ossec/etc/ossec.conf in the MANAGER_IP section and adding the IP address of the Wazuh server so that the process of connecting the honeypot to Wazuh can run properly. The last step taken in this automation is running the service from the Wazuh-agent so that the process of sending logs and events can run. The script for connecting honeypot and Wazuh Figure 11.

```
- hosts: honeypot
  become: true
  vars:
    wazuh: 54.179.178.79
  tasks:
  - name: Menambahkan key repo wazuh
    apt_key:
      url: https://packages.wazuh.com/key/GPG-KEY-WAZUH
      state: present
  - name: Menambahkan repo wazuh
    ansible.builtin.apt_repository:
      repo: deb https://packages.wazuh.com/4.x/apt/ stable main
      state: present
  - name: Update repo apt
    apt:
      update_cache: yes
  - name: install wazuh-agent
    apt:
      name: wazuh-agent
  - name: Konfigurasi WAZUH-manager IP
    command: /var/ossec/bin/agent-auth -m "{{ wazuh }}"
  - name: Update file konfigurasi ossec.conf
    command: sed -i 's/MANAGER_IP/{{ wazuh }}/' /var/ossec/etc/ossec.conf
  - name: Restart service wazuh-agent
    ansible.builtin.service:
      name: wazuh-agent
      state: restarted
```

Figure 11. Script Connecting Honeypot And Wazuh

3.7. Testing

1. Blackbox Testing

Testing is carried out to ensure the automation system created is in accordance with the design objectives and can facilitate the work of the system administrator. The total number of tests consisted of 6 tests on 6 functions as creating virtual machines, intrusion detection systems, honeypots, security information and event management and connecting intrusion detection systems, honeypots with security information and event management. All tests produce results that match the desired functionality.

2. User Acceptance Test (UAT)

The automation system that has been completed is then evaluated to find out how the results of meeting the needs of the automation system are, in this case using a questionnaire. Measurement of UAT results is carried out to determine whether the automation system can facilitate system administrators in managing and configuring security and monitoring network infrastructure. In this study the system was built with the aim of being implemented by the system administrator. Therefore, the selected UAT respondents are computer administrators at the Information Technology Unit of the National Cyber and Crypto Polytechnic. The results of the calculation of the UAT response data show that the implementation of infrastructure as code simplifies the management and configuration of security and monitoring of network infrastructure by 83.33%. The results of the data filling out the questionnaire resulted in a score of 250. Thus, it can be concluded that the implementation of infrastructure as code makes it easier to manage and configure security and monitor network infrastructure in the agreed or good area seen in Figure 12.

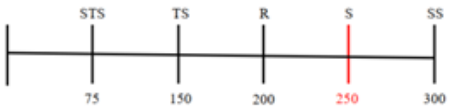


Figure 12. UAT Assessment Areas

4. CONCLUSION

Automation made using ansible makes it easier for system administrators because ansible is an agentless tool where the infrastructure to be automated or called remote nodes does not need to install any tools and only requires an SSH connection between control nodes and remote nodes. The automation that is made can make virtual machines, intrusion detection systems, honeypots, security information and event management to connect between the network infrastructure effectively and efficiently in terms of time because the process requires a short time. Based on the results of the UAT questionnaire, a score of 250 is obtained in the Agree area and the acceptance rate of infrastructure as code implementation for security automation and network infrastructure monitoring as a whole is 83.33%. In future research, automation can be carried out using other hypervisors such as Vmware vsphere, and Hyper-V

with a variety of operating systems and can also be tried to be implemented in cloud computing.

5. ACKNOWLEDGEMENTS

Thank you to Politeknik Siber dan Sandi Negara, who gave us opportunity to do this project on the topic Infrastructure as Code for Security Automation and Network Infrastructure Monitoring. This project gave us triggers to do more research.

6. DECLARATIONS

AUTHOR CONTIBUTION

Muhammad Hasbi, Agus Reza Aristiadi Nurwa and Dimas Febriyan Priambodo performed Conceptualization, Methodolog and Writing-Original draft preparation, Supervision, Reviewing and Editing. Wahyu Riski Aulia Putra performed Plugin programming, Prototyping and Integrating system.

FUNDING STATEMENT

Article publishing charges was provided by National Cyber and Crypto Polytechnic and research funding as coding and prototype was provided by all author.

COMPETING INTEREST

There is no conflict of interest for all author. Muhammad Hasbi is lecture and employee in STMIK Sinar Nusantara, Agus Reza Arisetiadi Nurwa, Dimas Febriyan Priambodo is lecture and employee in National Cyber and Crypto Politechnic, Wahyu Rizki Aulia Putra is graduated from National Cyber and Crypto Polytechnic and become employee in National Cyber and Crypto Agency.

REFERENCES

[1] T. Oktarina, "Media Pembelajaran Online Untuk Mendukung Belajar Pada Stebis Islam Darussalam," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 19, no. 2, pp. 329–338, 2020.

[2] I. P. Hariyadi and K. Marzuki, "Implementation Of Configuration Management Virtual Private Server Using Ansible," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 19, no. 2, pp. 347–357, 2020.

[3] K. Marzuki, R. Azhar, M. Mubiatma, J. Ismail Marzuki No, K. Cakranegara, K. Mataram, and U. Acid, "Otomasisasi Manajemen Vlan Intervlan dan Dhcp Server Menggunakan Ansible," *Jurnal Informatika & Rekayasa Elektronika)*, vol. 4, no. 2, pp. 171–180, 2021.

[4] H. Ahmetoglu and R. Das, "A comprehensive review on detection of cyber-attacks: Data sets, methods, challenges, and future research directions," *Internet of Things*, vol. 20, p. 100615, 2022.

[5] C. Kelly, N. Pitropakis, A. Mylonas, S. McKeown, and W. J. Buchanan, "A Comparative Analysis of Honeypots on Different Cloud Platforms." *Sensors (Basel, Switzerland)*, vol. 21, no. 7, apr 2021.

[6] A. Akhriana and A. Irmayana, "Web App Pendeteksi Jenis Serangan Jaringan Komputer Dengan Memanfaatkan Snort Dan Log Honeypot," *CCIT Journal*, vol. 12, no. 1, pp. 85–96, 2019.

[7] T. A. Cahyanto, H. Oktavianto, and A. W. Royan, "Analisis Dan Implementasi Honeypot Menggunakan Donaea Sebagai Penunjang Keamanan Jaringan," *Jurnal Sistem & Teknologi Informasi Indonesia*, vol. 1, no. 2, pp. 86–92, 2016.

[8] A. Khumaidi, "Implementation of Devops Method for Automation of Server Management Using Ansible," *Jurnal Transformatika*, vol. 18, no. 2, p. 199, 2021.

[9] I. P. A. E. Pratama, "Infrastructure as Code (IaC) Menggunakan OpenStack untuk Kemudahan Pengoperasian Jaringan Cloud Computing (Studi Kasus: Smart City di Provinsi Bali) Infrastructure as Code (IaC) Using OpenStack for Ease of Operation of Cloud Computing Network (Case Study," *Jurnal Ilmu Pengetahuan dan Teknologi Komunikasi*, vol. 23, no. 1, pp. 93–105, 2021.

[10] Nane Kratzke, "Infrastructure as Code — Cloud-native Programming."

[11] C. Siebra, R. Lacerda, I. Cerqueira, J. P. Quintino, F. Florentin, F. Q. B. da Silva, and A. L. M. Santos, "From Theory to Practice: The Challenges of a DevOps Infrastructure as Code Implementation," in *Proceedings of the 13th International Conference on Software Technologies*, no. Icsoft. SCITEPRESS - Science and Technology Publications, 2018, pp. 427–436. [Online]. Available: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006826104270436

[12] M. Artac, T. Borovssak, E. Di Nitto, M. Guerriero, and D. A. Tamburri, "DevOps: Introducing Infrastructure-as-Code," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, may 2017, pp. 497–498. [Online]. Available: http://ieeexplore.ieee.org/document/7965401/

[13] J. M. Kizza, *Guide to Computer Network Security - Fifth Edition*, 2017.

[14] A. O. Olagunju and F. Samu, "In Search of Effective Honeypot and Honeynet Systems for Real-Time Intrusion Detection and Prevention," in *Proceedings of the 5th Annual Conference on Research in Information Technology*, ser. RIIT '16. New York, NY, USA: ACM, sep 2016, pp. 41–46. [Online]. Available: https://dl.acm.org/doi/10.1145/2978178.2978184

[15] F. R. Hariawan and S. U. Sunaringtyas, "Design an Intrusion Detection System, Multiple Honeypot and Packet Analyzer Using Raspberry Pi 4 for Home Network," in *2021 17th International Conference on Quality in Research (QIR): International Symposium on Electrical and Computer Engineering*. IEEE, oct 2021, pp. 43–48. [Online]. Available: https://ieeexplore.ieee.org/document/9716189/

[16] R. M. R. Alan Dennis, Barbara Wixom, *Systems Analysis and Design*, 8th ed. John Wiley & Sons, Inc., 2021.

[17] BSSN-IHP, "Laporan Tahun 2020 Honeynet Project BSSN - IHP," Tech. Rep., 2021.

[18] L. Hochstein and R. Moser, *Ansible: Up and Running*, 3rd ed. O'Reilly Media, Inc., 2022.

[19] P. D. Sugiyono, *Metode Penelitian Kuantitatif, Kualitatif, dan R&D*, 3rd ed. Bandung: CV Alfabeta, 2021.