**Node Exporter via Ansible with Monitoring**

**Introduction**

**1. Prometheus -**

Prometheus is an open-source monitoring and alerting system designed for collecting, storing, querying, and alerting on time-series data.

**2. Node Exporter -**

Popular open-source software Node Exporter was created by the Prometheus community. It is a compact and effective Prometheus exporter made to gather different system-level metrics from platforms that operate similarly to Unix, such as Linux, Windows, and macOS. By exposing essential metrics for analysis, visualization, and alerting, Node Exporter enables you to keep an eye on the health and performance of your servers and workstations.

**3. Grafana -**

Grafana is an open-source monitoring and observability software that focuses on creating dashboards and data visualization. It is an effective tool for tracking the performance and well-being of systems, applications, and services because it is made to assist users in seeing and analysing their metrics and logs in real time.

**4. Ansible –**

You can automate the configuration, deployment, and maintenance of IT infrastructure, applications, and services with the open-source automation tool Ansible. It is intended to decrease manual intervention, streamline difficult tasks, and boost the effectiveness of IT operations.

**Module Explaination**

**Inventory File –**

[prometheus]

127.0.0.1

[node\_exporter]

127.0.0.1

[grafana]

127.0.0.1

In Ansible, the inventory file is used to specify the target hosts or groups of hosts on which you want to perform tasks, such as deploying software or running playbooks. The inventory file helps Ansible know where to apply the automation.

In my inventory files-

[Prometheus], [node\_exporter], and [Grafana] are group names and `127.0.0.1` represents a target host that belongs to that group. In my case, I am targeting the local machine.

Note – You can also add multiple hosts for other servers.

**Now we need to create roles –**

1. **Prometheus\_Node\_exporter :**

Command - ansible-galaxy init roles/prometheus\_node\_exporter

A Node Exporter role in Ansible is typically used to automate the deployment and configuration of Node Exporter, an agent that collects system-level metrics from servers and other hosts. These metrics are then made available in a format that can be scraped and stored by Prometheus, a popular open-source monitoring and alerting system.

Creating a Node Exporter role allows you to encapsulate all the necessary tasks, configurations, and files needed to install, configure, and manage Node Exporter on multiple hosts. This makes it easier to deploy Node Exporter consistently across your infrastructure and ensures that the required setup is maintained.

1. **Prometheus Role –**

Command - ansible-galaxy init roles/Prometheus

A Prometheus role in Ansible is a way to organize and encapsulate tasks, configurations, and files needed to install, configure, and manage Prometheus, an open-source monitoring and alerting system. Using a Prometheus role helps you create a modular and reusable approach to setting up Prometheus instances across different hosts or environments.

1. **Grafana Role –**

Command - ansible-galaxy init roles/Grafana

A Grafana role in Ansible is a way to organize and encapsulate tasks, configurations, and files needed to install, configure, and manage Grafana, an open-source platform for data visualization and dashboard creation. Using a Grafana role helps you create a modular and reusable approach to setting up Grafana instances across different hosts or environments.

Here's an example of how you might structure a role directory:

grafana\_role/

├── defaults/

│ └── main.yml

├── handlers/

├── tasks/

│ └── main.yml ├

── templates/

├── vars/

│ └── main.yml

└── README.md

* **defaults**: Contains default variables for the role.
* **files**: Could include configuration files, scripts, or other files needed for that role setup.
* **handlers**: Defines handlers, such as service restarts, triggered by tasks.
* **tasks**: Defines the main tasks for installing and configuring that role, setting up data sources, dashboards, and plugins.
* **templates**: Stores template files for Grafana configuration.
* **vars**: Contains variable definitions for the role.
* **README.md**: Documentation explaining the purpose and usage of the role.

**Steps for playbool.yml -**

The provided Ansible playbook consists of three main sections, each targeting different groups of hosts and applying specific roles. Let's break down the playbook step by step

---

- hosts: node\_exporter

  connection: local

  become: yes

  become\_user: root

  become\_method: sudo

  roles:

    - prometheus\_node\_exporter

In this section:

* `hosts: node\_exporter` specifies that the playbook tasks will be executed on the hosts belonging to the `node\_exporter` group. These are are hosts where you want to deploy and configure the Node Exported for Prometheus monitoring.
* `connection: local` indicates playbook will be run locally, not remotely over SSH which means that playbook is meant to configure the local machine where Ansible is running.
* `become: yes` indicates that the playbook tasks will be executed with escalated privileges (root)
* `become\_user: root` specifes the user to become when escalating priviledges.
* `become\_method: sudo` speicifies the method to use for privilege escalation.
* `roles:` defines the roles that will be applied to the hosts in the `Prometheus\_node\_exporter` group.

The playbook applies the same structure to the next two sections for **prometheus** and **grafana**:

- hosts: prometheus

  connection: local

  become: yes

  become\_user: root

  become\_method: sudo

  roles:

    - prometheus

- hosts: grafana

  connection: local

  become: yes

  become\_user: root

  become\_method: sudo

  roles:

    - grafana

These sections target the **prometheus** and **grafana** groups, respectively. They execute roles named **prometheus** and **grafana**, which are expected to contain the necessary tasks and configurations for deploying and setting up Prometheus and Grafana on the local machine (or the machine where Ansible is being run).

Roles are reusable units of work in Ansible that encapsulate tasks, variables, and configurations related to a specific component or function. The idea behind the playbook is to use roles to modularize the configuration and setup of different components (Node Exporter, Prometheus, and Grafana)s and apply them to appropriate host groups.

Overall, this playbook suggests a setup where Node Exporter, Prometheus, and Grafana are all deployed and configured on the same local machine where Ansible is running.

**Steps for Running the project**

1. Clone the repository.
2. Modify the IP address in inventory file accordingly for use case
3. Run the command - `ansible-playbook` -i inventory playbook.yml`

The command **ansible-playbook -i inventory playbook.yml** will execute an Ansible playbook using the specified inventory file (**inventory**) and playbook file (**playbook.yml**). The inventory file defines the target hosts or groups of hosts on which the playbook tasks will be executed. The playbook file contains a series of tasks and instructions to be carried out on those target hosts.

Note – If you want multiple hosts then below is the example of inventory file for multiple hosts –

[monitorserver]

db\_server   ansible\_host=<YOUR-DB-SERVER-IP>   ansible\_user=ec2-user  ansible\_ssh\_private\_key\_file=~/<YOUR-PEM-FILE>

[nodeservers]

server1  ansible\_host=<YOUR-WEB-SERVER-IP>  ansible\_user=ec2-user  ansible\_ssh\_private\_key\_file=~/<YOUR-PEM-FILE>

server2  ansible\_host=<YOUR-WEB-SERVER-IP>  ansible\_user=ec2-user  ansible\_ssh\_private\_key\_file=~/<YOUR-PEM-FILE>

Here you can add path to your public key in <YOUR-PEM-FILE> and ansible will use specified SSH key for authentication

Expected Output –

1. Ansible's standard output: During the execution of the playbook, Ansible will display output for each task it is running. This output includes task names, status (whether tasks were changed or not), and any additional debug or verbose information if enabled.
2. Progress updates: You'll see progress updates for each task as Ansible processes them. It will show which task is being executed and on which host.
3. Task results: For each task, you'll see whether the task was executed ("changed") or skipped because the system was already in the desired state ("ok").
4. Summary: Once the playbook execution is complete, Ansible will provide a summary that includes the number of hosts affected, the number of tasks executed, the number of tasks that resulted in changes, and the total runtime.

**Steps for Grafana –**

1. Go to url – localhost:3000
2. Add new Connection prometheus and enter url as – localhost:9090
3. Enter default username and password as “admin” for both.
4. Go to dashboard and click on new -> import and enter the dashboard id mentioned in <https://grafana.com/grafana/dashboards/1860-node-exporter-full/>
5. This will create a Grafana dashboard which consists of visualizations , panels , widgets that display metrics and data collected by Prometheus from target hosts

**Troubleshooting steps**

1. Check for IP in inventory file
2. Go to prometheus.conf.j2 in roles/prometheus/templates and add a list in target for job\_name: ‘node\_exporter’ containing your host ip and port number 9090 for prometheus

Note – if you have multiple target host then add the ip and port to the target list only.