Development of a system for monitoring hardware metrics

Aleksandar Hristov
Department "Information Technologies
in Industry"
Technical University of Sofia
Sofia, Bulgaria
ahristov@tu-sofia.bg

Abstract— **In this paper, a monitoring system that collects, stores and visualizes data from various sources has been developed. For this purpose, an infrastructure has been built using Vagrant for process automation and VirtualBox for virtualization hypervisor, Telegraf agents have been configured to collect hardware performance metrics such as CPU, memory, disk space and network bandwidth, InfluxDB has been installed and configured to store the collected from the hosts data and Grafana has been integrated as a visualization and analysis tool for dynamic data visualization.**

Keywords— IT infrastructure, monitoring, CPU, memory, disk space, network bandwidth, Telegraf agent, InfluxDB, Grafana, Oracle VirtualBox, Vagrant

I. INTRODUCTION

Nowadays, humanity is part of a modern era of digitization and rapid development of information technology. In this era, effective management and monitoring of IT infrastructure [3] is essential. Using a hardware metrics monitoring system provides an opportunity to monitor performance, identify potential issues, and optimize resources. The continuous growth of the IT infrastructure leads to an increase in complexity and the need for effective monitoring tools [1]. Implementing a hardware metrics monitoring system allows organizations to achieve higher reliability and security of their systems.

Determining which is the best combination of monitoring systems [4] is critical to the success of any IT infrastructure. This paper examines specific examples and reasons for using certain hardware metrics monitoring systems, including their functionality, reliability, and ease of integration.

First, it should be taken into consideration the ability of a monitoring system to collect and analyze real-time data. This is essential for quick detection of anomalies and taking the correct preventive measures. The second important aspect is the scalability of the system, which means the ability to add new devices and components to the system easy and without spending significant human and material resources. The third factor for choosing a monitoring system is the compatibility of the system with different hardware and software platforms, which ensures flexibility and integration in diverse IT environments.

As it was mentioned above, when determining which is the best hardware metrics monitoring system, it's important to consider a variety of factors, such as functionality, scalability, ease of use, and cost. Below is given a comparative analysis of some of the most popular systems.

Nagios [1] is one of the most wide-spread open source monitoring systems used to monitor networks, servers, services and even applications. It is able to monitor and manage the IT infrastructure, based on the configuration which resources to monitor and also has configurable alerts. Nagios is extremely flexible and expandable by using plugins, which allow adding new functionalities and resources to monitor according to the specific needs of the user. The main drawbacks of using Nagios are that the installation and configuration can be complex for inexperienced users, requiring specialized skills and setup time. Also, it has very limited visualization capabilities and cannot reach the level of graphical representation offered by more advanced solutions.

Zabbix is another popular IT infrastructure monitoring [1] and management platform that stands out for its flexibility and the variety of monitoring features. Zabbix provides monitoring capabilities for various types of resources, including servers, networks, applications, services, and other equipment. This software platform is also very flexible and supports automated processes for monitoring and managing the IT environment. The main drawbacks of using Zabbix are related to the initial setup and configuration, which require significant time and effort, especially for more complex monitoring and configuration scenarios, and the need for regular updates and maintenance to ensure stable and secure system operation.

SolarWinds [1] is another well-known system for monitoring and managing network devices, servers, applications and cloud services. It offers a wide range of monitoring and analysis tools that cover various aspects of the IT infrastructure. SolarWinds provides monitoring capabilities for devices from different types, including network switches, routers, servers, cloud services and applications, and enables centralized monitoring and administration of distributed IT systems through a single platform. It also has integration for automated processes and resource optimization tools, resulting in more efficient IT infrastructure management. However, using SolarWinds can be expensive due to licensing costs and the need of professional support. Also, this platform requires significant resources for data storage and processing, which can affect overall costs and performance.

979-8-3503-5500-0/24/$31.00 ©2024 IEEE

The purpose of present paper is to develop and implement an effective system for monitoring hardware metrics. The system should have the following functionality:

- collecting and storing data from end devices (hosts) about the CPU utilization, Random-Access Memory (RAM) and disk usage and network traffic in a central repository (database) for further processing;

- using a centralized model for management, communication and data storage in the system;

- a convenient and user-friendly web interface that will serve as the main panel for managing and monitoring the network and hosts.

II. ARCHITECTURE OF THE SYSTEM

The developed monitoring system consists of three main components, which are chosen after an analysis [4] of similar ones: Telegraf for data collection, InfluxDB for data storage and Grafana for data visualization. These components interact with each other as follows:

- Telegraf agent [6]: Collects hardware resources metrics from various hosts, e.g. servers, user PCs, etc. and sends this data to the InfluxDB for storing.

- InfluxDB [5]: Stores the collected data about the CPU utilization, memory and disk usage and network traffic;

- Grafana [7]: Uses the data from InfluxDB to create visualizations that can be used for better understanding of the user and also offers a functionality to trigger alarms if some value is critical.

Telegraf has the ability to use various input plugins to collect data. For the developed system, these data include following metrics:

- CPU performance indexes;

- RAM usage;

- Disk space usage and disk operations;

- Network traffic and bandwidth usage.

InfluxDB stores the data collected for those metrics in a structured form allowing for quick access and analysis. Key features of InfluxDB include real-time data writing and reading, complex query support via InfluxQL [5] and Flux languages, data aggregation and transformation capabilities.

Figure 1 shows the dataflow of the developed system and operation of the Telegraph agent.

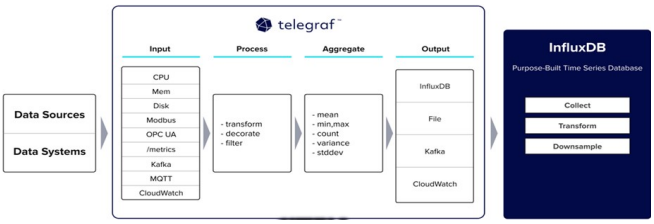

Fig. 1. System structure and dataflow

Grafana [7] is used as visualization tool for the data that is stored in InfluxDB in order to create interactive graphs and dashboards. Grafana's main features include creating graphs and dashboards, supporting various data sources, and functionality for creating alarms and notifications.

Vagrant [8] is a management tool for virtual environment that makes it easy to configure and manage development virtual machines (VMs). It allows developers to create and manage virtual environments with minimal effort via a plain text file configuration.

Oracle VM VirtualBox [9] is an open-source type 2 hypervisor, which means it requires an existing OS to be installed on the hardware. It allows users to create and manage virtual machines with different operating systems by running them on a host operating system. Virtual machines are created as isolated virtual environments that can be used for development, testing, training, etc.

For the implementation of the developed system, Vagrant with VirtualBox was used to create virtual machines and to automate the configuration through provisioning scripts. The configuration of the virtual machines is as follows: VM1 – Telegraf agent and Grafana, VM2 – Telegraf agent and InfluxDB, VM3 - Telegraf agent, VM4 - Telegraf agent.

A. Collecting and analyzing hardware metrics via Telegraf agent

This section describes the process of collecting and analyzing hardware metrics using Telegraf agent [6]. Telegraf is a metrics collection agent that supports multiple input and output plugins. Below are the plugins used in the Telegraf agent configuration for the implementation of the developed system:

- inputs.cpu is used for collecting CPU performance index data by continuously monitoring various aspects of CPU activity such as CPU usage, load distribution between the CPU cores, cache usage, task execution time and other relevant metrics;

- inputs.mem is used for collecting data about the RAM usage by continuously monitoring various aspects of RAM, including the total amount of RAM available, usage by active processes, cache and buffer usage, and free memory percentage;

- inputs.disk is used for collecting data about the hard disk by continuously monitoring various aspects of the disk, including the total amount of free and used disk space, read and write speeds, and the space occupied by individual file systems;

- inputs.net is used for collecting data about network interfaces by continuously monitoring all active network interfaces, including metrics such as input and output data transmission speed (bps), total number of sent/received packets, errors and dropped packets. These metrics are useful for analyzing network load and extracting network performance statistics.

In the Telegraf configuration file, various parameters and settings can be set for all of the above plugins, such as data collection interval, accuracy, how metrics are presented, and other parameters to suit the specific user needs. Using those plugins, Telegraf generates data that is then sent to a data storage system such as InfluxDB. These data are then visualized and analyzed using tools such as Grafana.

B. InfluxDB Database

InfluxDB [5] is a modern time-series database designed to collect, store and analyze large volumes of temporal data. In this section is described the process of setting up users,

organizations, and buckets in InfluxDB. Users in InfluxDB have different access rights to the data and different administrative functions of the database. First, a user must be created using the CLI command:

influx user create -n <username> -o <organization> -p <password>

The command above is also used to manage user rights trough the organizations. It is important to enable authentication and authorization to protect data. Organizations in InfluxDB are logical units that group together users and resources such as Buckets and Dashboards. Organizations help structuring and managing data access. The command below is used for creating a new organization:

influx org create -n <organization_name>

Buckets in InfluxDB are logical data containers. They are analogous to databases in other database management systems and are used to organize time series [2]. The command below is used for creating a new bucket:

influx bucket create -n <bucket_name> -o <organization> -r <retention_period>

Figure 2 shows the path of the data (metrics) that are sent from the hosts to the InfluxDB and through the Bucket (COLLECTION) can be extracted by the search panel of the Grafana web interface.

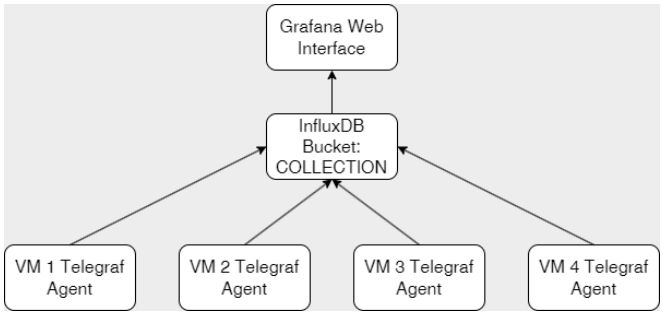

Fig. 2. Performance indexes and datapath

III. Experiments and results

Grafana [7] is a powerful data visualization and analysis tool that offers various functionalities for creating interactive graphs and dashboards, notifications and alarms.

This section is dedicated to Grafana's integration with the InfluxDB. Before starting visualizations, Grafana must be configured to use InfluxDB as a data source. The steps to add a data source are given below:

1. Open the web interface of Grafana and choose „Data Sources”;

2. Click the „Add data source” button and choose InfluxDB;

3. In the URL field, input the address of the InfluxDB server;

4. The required authentication data must be given;

5. Click the „Save & Test” button to check if the connection to the database is successful.

Once InfluxDB is configured as a data source, a dashboard has to be created. The steps for creating and configuring a dashboard are as follows:

1. For creating a new dashboard: click the "Create" button and select "Dashboard. Then, a new panel is added to the dashboard by pressing "Add new panel;

2. For Panel configuration select the type of chart to be used and in the Query section select InfluxDB as the data source. Then a query has to be written to InfluxDB. Due to the limited size of this paper, the quarries that are used for creating the system are given in the author’s repository in github [10].

3. The visualization is adjusted according to the user's needs.

Fig. 3. Visualization of the collected data for VM1 in Grafana

The last step is to configure Grafana for data analysis. One of the main features of Grafana is the ability to analyze data in real time. Various filters and time ranges can be used to gain a better understanding. Time filters are at the top of the dashboard and are used to change the range of data that is used for creating the visualizations.

The created dashboard that is used to monitor the data that VM 1's Telegraph agent intercepts and stores in the InfluxDB is shown on fig. 3.

One of the most important functionalities of Grafana is creating alarms and notifications. Alarms can be set to trigger when certain thresholds are reached. For example, if CPU usage exceeds 80%, Grafana can send a notification through email or Discord message. An example alarm that is used in the developed system is given below:

alert{
query = Acondition = avg() of query(A, 5m, now) > 80
timeRange = 5m
interval = 1m
}

Data collecting from the hosts is achieved via Flux Queries. Due to the limited volume of the paper, in [10] the author of the paper provided 4 different queries for the different types of metrics that are monitored by the developed system. These are the queries for CPU usage percentage, used nodes, free disk space and network traffic/bandwidth.

IV. Conclusion

In this paper, a monitoring system that successfully collects, stores and visualizes data from various sources has been developed. Key results include:

- An infrastructure has been built (multi-host virtualized environment) using Vagrant for process automation and VirtualBox for virtualization hypervisor. This provides a stable and controlled test environment;

- Telegraf agents have been configured on all VMs to collect hardware performance metrics such as CPU, memory and disk space usage and network bandwidth;

- InfluxDB has been installed and configured as a database to store the collected from the hosts data. Buckets are used to organize data and to manage access of user accounts. Organizations are used to structure users and access rights.

- Grafana has been integrated as a visualization and analysis tool with the InfluxDB and dynamic data visualization dashboards have been created.

The developed monitoring system has several key benefits:

- The ability to collect and visualize real-time hardware metrics gives administrators better visibility and control over networks and systems;

- Setting up alarms and notifications in Grafana enables proactive resource management and the ability of preventing potential problems before they become critical incidents;

- The developed system provides tools for detailed performance analysis and identification of potential bottlenecks in the infrastructure.

There are several possibilities for future development and improvement of the system that are given below:

- expansion of the metrics (inclusion of additional data sources, such as application logs, which will provide a more comprehensive picture of the monitored networks);

- integration with other monitoring and management tools, such as Prometheus and ElasticSearch, to extend functionalities and improve analytical capabilities;

- using automation and orchestration tools, such as Ansible or Kubernetes, to more easily manage system configurations and scaling.

Acknowledgment

The author would like to thank the Research and Development Sector at the Technical University of Sofia for the financial support.

References

[1] Julian, M. Practical Monitoring, O'Reilly Media, 2017, ISBN: 9781491957318

[2] Atwan, T. A. Time Series Analysis with Python Cookbook, Packt Publishing, 2022, ISBN: 9781801075541

[3] Turnbull, J. The Art of Monitoring, Turnbull Press, 2014

[4] Hristov A., R. Trifonov, System for simulating cyberattacks and data acquisition for network traffic and usage of processor and memory of hosts, Proceedings of International Scientific Conference "Engineering. Technologies. Education. Safety, 2022", Publisher: Scientific technical union of mechanical engineering “Industry - 4.0”, ISSN 2535-0315, pp. 39-42

[5] Hermans, K. Mastering InfluxDB Database: A Comprehensive Guide to Learn InfluxDB Database, Cybellium Ltd, 2023, ISBN: 979-8867766450

[6] Documentation of Tefegraf online available: https://docs.influxdata.com/telegraf/v1/

[7] Documentation of Grafana online available: https://grafana.com/docs/

[8] Documentation of Vagrant online available: https://developer.hashicorp.com/vagrant/docs

[9] Documentation of Oracle Virtualbox online available: https://www.virtualbox.org/wiki/Documentation

[10] Hristov, A. IT infrastructure monitoring system online available: https://github.com/sashkinaaa/monitoringSystem