Prometheus+Grafana-Based Unified Operations and Maintenance Monitoring for Xinhua All-Media News Service Platform (Postprint)

Authors: Zhong Yingjiong

Date: 2023-10-08T00:00:00+00:00

Abstract

To achieve unified operation and maintenance monitoring for Xinhua News Agency' s content supply platform, the Xinhua All-Media News Service Platform built on Docker (which serves as Xinhua News Agency' s content supply platform, hereinafter referred to as the All-Media Platform) has become the content receiving platform for domestic and overseas users of Xinhua News Agency. Monitoring the platform' s operational status constitutes one of the critical guarantees for the stable and secure operation of the All-Media Platform.

This article primarily introduces the methodology for implementing system monitoring of the Docker-based All-Media Platform utilizing open-source tools Prometheus and Grafana.

This paper first introduces the technical fundamentals of the aforementioned two open-source tools and Docker microservices deployment, then elaborates on the technical architecture of the monitoring system, thereby illustrating the strategy, implementation plan, and process for constructing an online enterprise-level operation and maintenance monitoring system using these tools, achieving comprehensive monitoring capabilities for interface visualization and alert notification.

Through the construction of an enterprise-level unified operation and maintenance monitoring platform for the Xinhua All-Media News Service Platform based on open-source frameworks including Prometheus and Grafana, the feasibility of the design scheme has been validated, the operational efficiency of maintenance personnel has been enhanced, and system stability has been ensured.

Full Text

Abstract

To achieve unified operations monitoring for Xinhua' s content delivery platform, the Xinhua All-Media News Service Platform (hereinafter referred to as the "All-Media Platform")—built on Docker and serving as Xinhua's domestic and international content delivery system—requires robust monitoring as a critical guarantee for stable and secure operation. This paper introduces the methodology for monitoring the Docker-based All-Media Platform using the open-source tools Prometheus and Grafana. We first present the technical essentials of these two open-source tools and Docker microservice deployment, then elaborate on the technical architecture of the monitoring system to illustrate the strategy, implementation plan, and process for building an enterprise-grade online operations monitoring system using these tools, achieving integrated monitoring effects for interface display and alert notifications. By constructing an enterprise-level unified operations monitoring platform for the All-Media Platform based on Prometheus, Grafana, and other open-source frameworks, we validate the feasibility of the design scheme, enhance the work efficiency of operations staff, and ensure system stability.

Keywords: Docker; Prometheus; Grafana; microservices; Xinhua All-Media News Service Platform

2. Overview of Prometheus and Grafana

This section first introduces Docker, Prometheus, and Grafana, then explains the microservice architecture of the Docker-based All-Media Platform, proposes the design scheme and implementation process for enterprise-grade system monitoring using Prometheus+Grafana, and finally presents partial visualization effects.

Prometheus is an open-source monitoring and alerting system and time series database (TSDB) developed by SoundCloud. Written in Go, it represents the open-source version of Google's BorgMon monitoring system. In 2016, the Cloud Native Computing Foundation (CNCF) under the Linux Foundation adopted Prometheus as its second major open-source project. Prometheus and Heapster (a Kubernetes sub-project for cluster performance data) together provide more comprehensive functionality, with Prometheus performance sufficient to support cluster deployments at tens of thousands of nodes.

Grafana is an open-source, powerful visualization tool for monitoring and analysis, featuring fast and flexible client-side charts and modular tools. Its dashboard plugins offer numerous visualization methods for metrics and logs, including rich official library resources such as heatmaps, line charts, and various display formats that render complex data elegantly. Grafana supports many time series databases as data sources, including the Prometheus system discussed in this paper.

4. All-Media Platform System Based on Containerized Microservice Architecture

4.1 Introduction to Microservices

Microservice architecture is a software development framework for internet application services, evolving from Service-Oriented Architecture (SOA). It advocates dividing monolithic applications into a set of small services that coordinate and cooperate with each other. This approach has become a popular direction and primary choice for system design and technical implementation in the industry.

4.2 Transition from Traditional to Microservice Development Framework

The original All-Media Platform in our organization was built using the open-source Dubbo framework—large and complex, making agile development and iterative deployment cumbersome, particularly difficult during upgrades and rollbacks. The current All-Media Platform has transitioned from traditional service architecture to a Spring Cloud-based microservice architecture, utilizing local Consul clients for service registration, discovery, and consumption with Consul servers. During registration, services report their IP addresses and ports, then send health checks at regular intervals. When consumption is required, clients first obtain a temporary table containing IPs and ports from the Consul server before accessing the actual routes.

4.3 Business Service Deployment Architecture

To achieve global site deployment, the business service architecture design fully considers business expansion and changes. Key components include: gateway (application service routing gateway), doc-view (manuscript viewing), doc-server (manuscript service), auth (user authentication and authorization), management (backend resource management), consul server (service registration and discovery), and manuscript ingestion. The business deployment architecture is shown in [Figure 3: see original paper].

4.4 Containerized Deployment Implementation

Based on the subdivision of microservice modules, we implemented automated operations capabilities for modular packaging and CI/CD. Our project team utilized Docker to conveniently deploy application services in a "containerized" manner, packaging all required environments within images—truly "build once, run anywhere." For scalable and clustered management of Docker containers, we employed Kubernetes (K8s), Google' s container cluster management system, which primarily includes automated deployment, automatic scaling, and container maintenance management. In this project, K8s 统一管理各个应用 Docker 容器，根据业务需求和访问情况动态扩展，确保系统服务的稳定性、安全性和可靠性。

5. Building an Enterprise-Grade Operations Monitoring Platform for the All-Media Platform

5.1 Functional Architecture

The platform comprises four main functional modules: metric data collection, metric storage, visualization display, and alert management. The collection module gathers data from all metric interfaces and stores the time series data in the metric storage TSDB for long-term retention. The visualization module utilizes this time series data for various forms of visual presentation, while the alert management module matches rules against the time series data and triggers grouped, deduplicated notifications when rules are violated.

5.2 Technical Architecture and Implementation

The platform is built on open-source technologies and self-developed modules, running on a containerized environment at the 底层. Key open-source technology selections include:

- **Prometheus**: As detailed in Section 3, Prometheus serves as the secondary scraping service in this system. Its Web UI facilitates debugging, updates, and maintenance for secondary scraping personnel.
- **Grafana**: As detailed in Section 3, Grafana queries and analyzes time series data from the TSDB for visualization.
- **VictoriaMetrics**: A component within the VictoriaMetrics suite that serves as the centralized metric storage. It is a high-availability, low-consumption, scalable time series database for long-term storage of Prometheus-standard time series data.
- **vmagent**: The metric collection component in the VictoriaMetrics suite, which can collect massive amounts of time series data more efficiently than Prometheus with lower resource consumption.
- **vmalert**: The alerting component in the VictoriaMetrics suite, which executes a series of given rules (based on MetricsQL, a superset of PromQL) and sends alert information to the Alertmanager component.
- **Alertmanager**: The alert notification component that receives alerts from vmalert and sends them through various channels, performing deduplication, noise reduction, grouping, and policy routing.
- **lanxin-gateway**: The BlueMessage gateway component that receives alerts from Alertmanager, preprocesses and formats them, then sends alert messages to corresponding BlueMessage groups via group message APIs.

5.3 Deployment Architecture

[Figure 6: see original paper] illustrates the deployment architecture of the operations monitoring platform. We plan to deploy one or more servers at four major content delivery sites globally and at the Beijing headquarters to host monitoring components. Functionally, the architecture divides into: secondary scraping

servers deployed at four application service sites to collect local monitoring information, which is then sent to the headquarters for unified aggregation and management; and centralized metric collection servers and core servers deployed at the Beijing headquarters.

5.3.1 Core Service Layer As the top layer in [Figure 6: see original paper], the core service layer is primarily responsible for time series data storage, alert rule computation (alert service), alert matching operations, alert message grouping and distribution (Alertmanager), and visualization display (Grafana). Specifically, it receives formatted and standardized data from the data collection layer, analyzes and filters it, stores it uniformly in the time series database, and provides visualization templates for users while generating alert data based on business requirements.

The middle primary scraping layer and bottom secondary scraping layer in [Figure 6: see original paper] mainly collect server host data, infrastructure component data, and monitored application service data, standardizing the collected data for Prometheus service collection interfaces. Specifically: - **Primary scraping layer**: Responsible for pulling metrics from all secondary scrapers in its region and storing this data in the core server' s time series database. Primary scrapers also push metric data to the headquarters' core servers to ensure data availability. - **Secondary scraping layer**: Responsible for scraping metrics within its managed service site, including system data and infrastructure component metrics from all physical and virtual machines, as well as critical application service interface metrics from the four All-Media Platform sites. The collected metrics must sufficiently ensure the security, reliability, and stability of business service states.

5.4 Implementation Process

5.4.1 Data Collection Configuration Unified operations monitoring cluster services are built at the four sites and headquarters aggregation point, divided into headquarters aggregation, primary scraping, and secondary scraping. At each site, exporters are installed to collect basic data, including CPU, memory, filesystem, disk, network, TCP connections, and key metrics such as CPU mode seconds, 5-minute average load, total/free/available memory bytes, filesystem bytes, and components like MySQL, MongoDB, Nginx, Elasticsearch, and Redis.

Application metric collection programs are deployed at each site, along with white-box/black-box probes and log analysis services, transforming data into formats recognizable by Prometheus and extractable by upper layers.

5.4.2 Data Display Configuration After logging into the headquarters server and installing Grafana (or at branch sites for local data viewing), administrators connect Grafana to the time series database data source via the Web interface. Appropriate display modules are selected (or obtained from the

official website if unavailable), and Json files or module IDs are imported into Grafana.

By comprehensively analyzing business display requirements and effects, various data types are selected—including infrastructure and application service metrics—and displayed on Grafana' s Web interface.

5.4.3 Alert Rule Configuration Grafana' s alerting is panel-based, with each panel configured independently for alert rules, trigger conditions, notification channels, and content. The configuration process involves: 1. Specifying notification channels and modifying the Grafana configuration file (grafana.ini) 2. Logging into the Grafana Web interface, accessing the settings area, configuring alert reception channels, and setting thresholds

6. Overall Interface Effects

After configuring the Grafana service address and port, users can log into the Grafana Web service, connect to the corresponding time series database, and display the collected formatted data on the operations monitoring dashboard. By strategically selecting different display colors and methods, data is presented in a planar, graphical, and easily readable format, enabling operations staff to monitor the real-time status of the infrastructure and network environment, and obtain system status and alarm information promptly.

6.1 All-Media Platform Monitoring Overview

[Figure 7: see original paper] displays the overall status of All-Media Platform business, ports, interfaces, and processes. If a module turns red, it indicates an alarm condition. Hovering over the hyperlink icon at the top-left of a panel reveals a drill-down link to detailed secondary monitoring pages. The lower-left section shows connectivity test details for the four sites of the New Content Delivery 2.0 system. Clicking any red module provides access to detailed secondary monitoring pages.

6.2 All-Media Platform Secondary Monitoring Details

[Figure 8: see original paper] and [Figure 9: see original paper] display process service status and port service status within the business system, while [Figure 10: see original paper] and [Figure 11: see original paper] show the overall infrastructure and network environment. Colors change to yellow, red, or other statuses to help operations personnel quickly understand system conditions.

Simple and intuitive monitoring displays are essential tools for system operations, while correct and timely alerts form the foundation of service stability. As microservice architecture concepts are applied to the All-Media Platform' s production environment, the open-source Prometheus+Grafana combination—characterized by simplicity, stability, reliability, and scalability—has become

the preferred choice for building enterprise-grade operations monitoring platforms. The design scheme and implementation details presented in this paper have effectively enhanced the efficiency and capabilities of All-Media Platform operations staff, further ensuring stable and reliable system operation.

References

[1] Micro Enjoyment. Introduction to Docker [EB/OL]. https://baijiahao.baidu.com/s?id=1692361731135557771 2021-02-22/2022-12-23.
[2] Linux Person. Prometheus Introduction: Open-Source Monitoring System and Time Series Database [EB/OL]. https://ywnz.com/linuxysjk/2287.html. 2018-07-14/2022-12-21.
[3] Introduction to Visualization Tool Grafana [EB/OL]. http://www.360doc.com/content/19/0711/10/1317564 2019-07-11/2022-12-22.
[4] Han Xiao, Li Jieyuan. Exploration of News Production System Optimization Based on Microservice Architecture [J]. China Media Technology, 2021(2): 62-63.
[5] Sun Bo. Discussion on Microservice Architecture, Docker, and Kubernetes [J]. Modern TV Technology, 2022(2): 100-103.
[6] Shao Ke, Cai Guohua, Wan Guolei. Deployment Scheme of China Search Kubernetes Application Platform [J]. China Media Technology, 2019(5): 113-117.

Author Profile: Zhong Yingjiong (1981-), male, from Zhuji, Zhejiang, holds a master' s degree, works as a senior engineer at the Communication Technology Bureau of Xinhua News Agency.
(Responsible Editor: Zhang Xiaojing)

Note: Figure translations are in progress. See original paper for figures.