ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV031

THE DEPLOYMENT TECHNOLOGY OF EPICS APPLICATION SOFTWARE BASED ON DOCKER

R. Wang†, Y. H. Guo, B. J. Wang, N. Xie, IMP, Lanzhou 730000, P. R. China

Abstract

StreamDevice, as a general-purpose string interface device's Epics driver, has been widely used in the control of devices with network and serial ports in CAFe equipment. For example, the remote control of magnet power supply, vacuum gauges and various vacuum valves or pumps, as well as the information reading and control of Gauss meter equipment used in magnetic field measurement. In the process of on-site software development, we found that various errors are caused during the deployment of StreamDevice about the dependence on software environment and library functions, which because of different operating system environments and EPICS tool software versions. This makes StreamDevice deployment very time-consuming and labor-intensive. To ensure that StreamDevice works in a unified environment and can be deployed and migrated efficiently, the Docker container technology is used to encapsulate its software and its application environment. Images will be uploaded to an Aliyun private library to facilitate software developers to download and use. At the same time, some technical implementations, to ensure the safety of containers and host, for communication security between containers and system resource management should be taken on our host. In the future, the large-scale efficient and stable deployment of StreamDevice software can be realized by pulling images on the server installed with Docker, avoiding a lot of repetitive work and greatly saving the time of control technicians.

INTRODUCTION

CAFe equipment, the CiADS superconducting linear accelerator prototype, was built in 2018 to validate the superconducting linear accelerator for CiADS with 10 mA continuous beam current. The equipment consists of the ion source, low-energy transmission section, RFQ, medium-energy transmission section, superconducting linear accelerator, high-energy transmission section, and beam terminal collector. The equipment layout is shown in Fig. 1. The overall parameters of the equipment include the beam intensity of 10mA, the beam energy of 25-40MeV, the RF frequency of 162.5MHz, as well as the temperature of 4.5k [1-3].

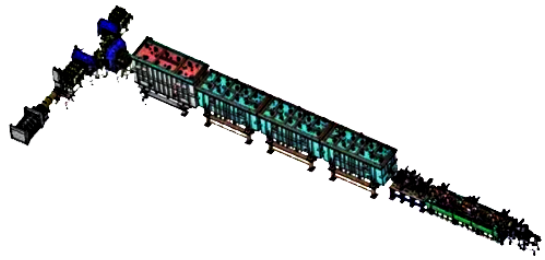

Figure 1: Layout of CAFe superconducting proton linear accelerator.

At present, EPICS is mostly used in the CAFe equipment control software of the Institute of Modern Physics, Chinese Academy of Sciences. The experimental physics and industrial control software adopt the standard distributed control system model and run stably. The device driver is developed with unified system architecture. It has gradually replaced LabVIEW as the mainstream software architecture and development tool for the accelerator control system in China and abroad. StreamDevice module is needed to realize serial communication equipment and drive control, which has the interface of the asynchronous drive module (ASYN) and can provide equipment support for EPICS [4].

With the upgrading of CAFe equipment, the servers have also been updated. In practice, it is found that there are often encountered unknown errors in the installation of EPICS on each new server. A lot of manpower and time will be paid to find the cause of the problems and solve methods. But the solutions do not possess universality. In order to improve the deployment efficiency and reduce the difficulty of application migration, Docker [5] is used to encapsulate EPICS + StreamDevice to assure the unification of the operating environment on different machines. At the same time, a series of security problems existing in the current container should be optimized and improved.

FUNCTIONALITY AND REQUIREMENTS

An asynchronous drive module (ASYN) interface was integrated with StreamDevice, which supports serial ports (RS485, RS232), IEEE-488 (GPIB or HP-IB), and Ethernet interfaces. It can realize the communication between serial ports and Gaussian power supply. The communication rules are specified by writing protocol files, and PV information is defined by DB files [6]. The result of the EPICS reads and sets the equipment information was achieved. Because the StreamDevice, a model of EPICS, was used to develop of I/O IOC for power and Gauss meter.

StreamDevice can analyze web pages by regular expressions, as well as read this important information in real

† wangrui@impcas.ac.cn

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV031

time in the web control interface to learn whether the archiving system is running. After setting PV, CA Server performs data archiving by Archive Engine. And these data should be submitted by Achieve Export to browse. Finally, the information is displayed in the human-machine interface on Web Apps [7]. The process is shown in Fig. 2.

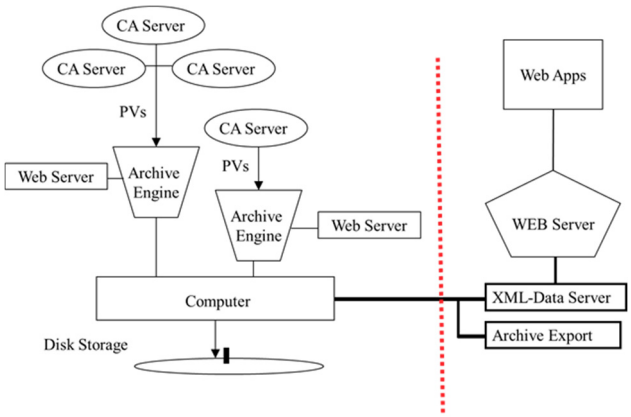

Figure 2: StreamDevice is used to monitor the running status of your app and parse web pages.

INTEGRATION OVERVIEW

EPICS has been widely used in the field of CAFe equipment control. The application based on EPICS has good portability, scalability, reusability, and maintainability. EPICS can be divided into three parts from top to bottom: the upper layer is the operator interface OPI, which can complete the human-machine interaction (HMI) process. The middle layer is the input and output controller, referred to as the IOC. The lower layer is channel protocol CA, CA protocol is divided into two parts: CA client and CA server [8]. The servers running IOC can be accessed via TCP/IP by the client. ASYN provides driver support for communication with hardware devices. StreamDevice, a generic EPICS device, can control devices by sending and receiving strings [9].

Docker, a lightweight application container deployment framework, can package, publish, and run any applications [10]. Application migration can be realized by firstly packaging EPICS+StreamDevice with Docker, secondly building images and sending them to the remote registry (docker hub), finally pulling and running images on deployed servers. The overall framework of the system is shown in Fig. 3.

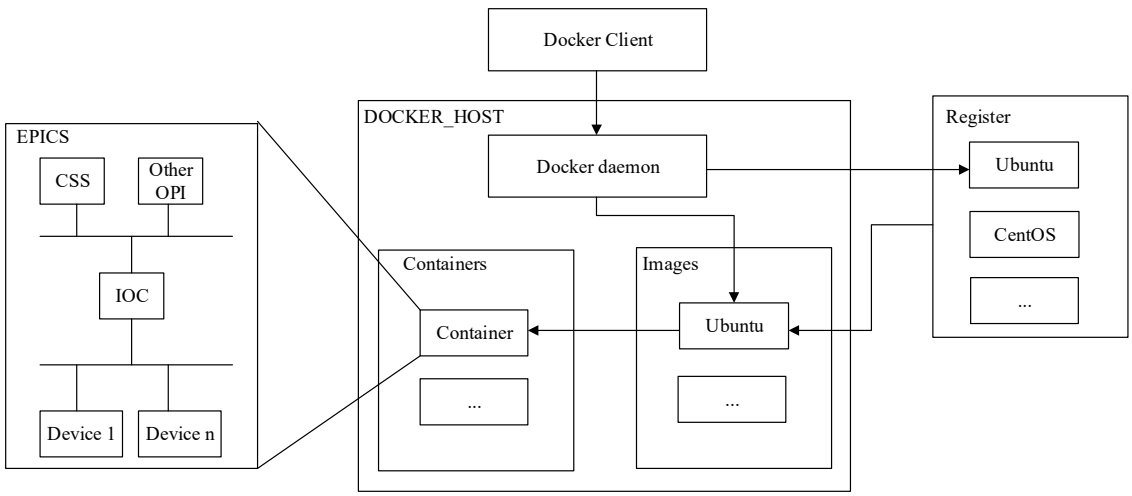

Figure 3: The overall framework of the system.

DOCKER CONTAINER TECHNOLOGY AND SECURITY

Docker

Docker is considered as an Internet container and will be the future standard for PaaS [11]. Before using Docker to package EPICS + StreamDevice, if the server is updated or the server is abnormally collapsed, these applications need to be redeployed. The deployment process is complex and may encounter unknown problems, which takes a lot of manpower and time. Packaging all effective loads through Docker enables consensus migration between almost any server, which greatly simplifies deployment and maintenance. The official Ubuntu 18.04 is used as the basic image. The container should be constructed from the image, which installs EPICS + StreamDevice in the container. After all, the container will be committed as a new image.

Images Security Restrictions

The official image registry allows all Docker hub users to visit. All the application projects on it are open source, which allows all users to pull and change. This openness breaks the technical barrier and facilitates project delivery. At the same time, there are some immeasurable security risks of data breaches and image misappropriating [12]. The private image stowage was built through Aliyun to increase the authentication mechanism when attempting to pull the image. This method can limit the pull permissions, as well as ensure the security of the image.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV031

Container Resource Security Mechanism

The container is different from the virtual machine. There is no independent resource allocation to make it impossible to isolate resources at the system kernel level. Since the Docker container shares the operating system kernel with the host, there is a risk that the container is controlled by the attacker to obtain the access rights of the host file. Or the attacker obtains the root permission by illegal means to control the host and other containers on the host. It will result in the escape of the container, or even that the Docker container exhausts the host to make the host or other containers pause or die [13]. In summary, it is necessary to limit the resources of container access to host through Cgroup and SELinux, and the details will be described in the test section.

Container Network Security Mechanism

Docker's default network communication mode is bridge mode, which will virtualize a docker0 bridge on the host computer, and connect all the containers used in this mode on the host computer to the bridge. At this time, all Docker containers can reach each other, which makes attackers may threaten the safety of the host computer and the container through broadcast storms, sniffing, ARP fraud, and other attacks [14]. When Docker containers provide services alone (such as EPICS + StreamDevice in this paper) without requiring multiple Docker containers to form micro-services, it is necessary to prohibit communication between containers, which can be achieved by setting the parameter dockerd-icc.

PERFORMANCE

Image Testing

The basic image used in this paper is the official image of Ubuntu 18.04. Based on this image, the dockerfile [15] file is written. The basic image, the author's signature, the working path, and the container number are specified. The host directory file is mounted and the environmental path is edited. Each step can be seen as an additional layer in the original image.

After entering the container, EPICS is installed and then Asyn and StreamDevice are deployed. The deployment results and PV display results are tested. EPICS + StreamDevice in Fig.4 has been successfully deployed in the container. Committed the successfully tested container as a new image and pull the packaged image on the required server.

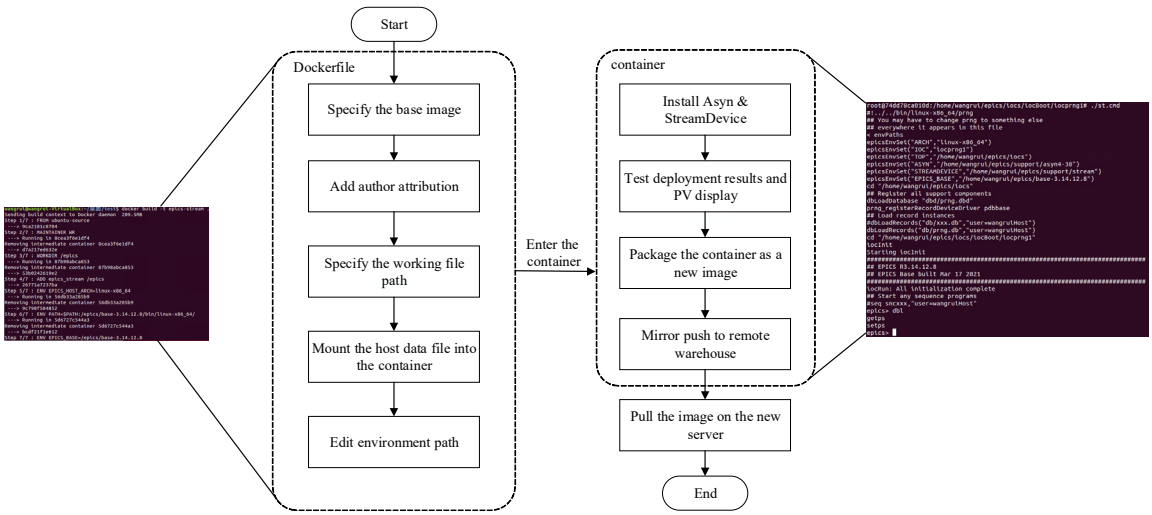

Figure 4: Containers are built by dockerfile and EPICS was deployed successfully with the PV is displayed.

Mirror Access Authentication Test

By establishing a private image stowage under the Aliyun platform, the problem of image pulling authentication can be solved. First Aliyun account should be logged in to establish an image stowage and create a namespace. Then the image Tag that needs to be uploaded should be changed on the host running docker. The image is uploaded through the push command. When the image needs to be pulled, the remote Aliyun image stowage needs to be logged in through docker, as well as the pull command can be used only after the authentication is completed, which limits the pull of private image by users who cannot complete the login authentication to a certain extent. The process is shown in Fig. 5.

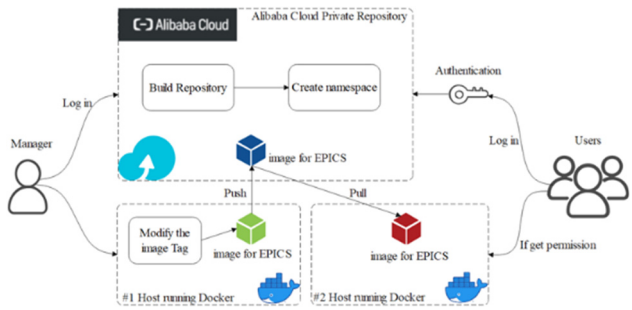

Figure 5: Aliyun private repository image access authentication.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV031

Resource Limit Testing

Cgroups can limit resource items such as CPU, memory, and disk I/O speed of the Docker container to avoid a single container running out of hardware resources on the host. Table 1 takes memory as an example to show the difference of system resources occupied by containers under the condition of whether to open Cgroups constraints. After the Cgroups memory limit was opened, the process is killed if the container actually uses more memory than the limit. If Cgroups was not used to limit memory, the container will occupy too much memory of the host, which may lead to the death of other containers. Experiments show that the container takes up five times more memory than using Cgroups. Finally, Docker's mandatory access control of host resources is implemented by starting the SELinux mechanism.

Table 1: Cgroups Memory Limit Comparison

Status	System total memory(MB)	Set limited memory(MB)	Memory usage(MB)	Percentage of total memory
Use Cgroups	2048	200	1.08	0.05%
Do not use Cgroups		—	5.14	0.25%

Network Communication Testing

The EPICS + StreamDevice container used in this paper provides services separately, so it should be forbidden to communicate with other containers running on the host computer. In the test environment, the IP of the EPICS + StreamDevice container is 172.17.0.2, and that of the Ubuntu container is 172.17.0.3. It is necessary to set dockerd-icc = false to prohibit communication between them, as shown in Fig. 6.

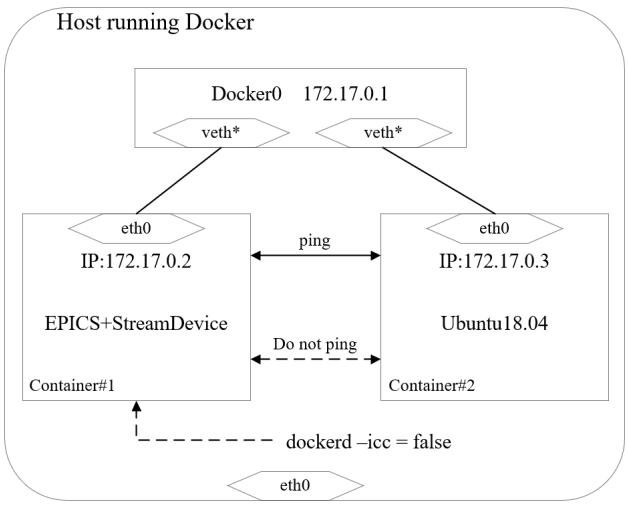

Figure 6: All containers running on the same host are in the same network segment, and if they are not restricted they can ping through each other, restarting the Docker service by limiting the icc parameter can achieve the purpose of prohibiting communication between two unrelated containers.

SUMMARY

The packaging of EPICS application by Docker can effectively reduce the deployment difficulty of EPICS. It also reduces the cost of application migration and solves a series of problems caused by the difference between the test environment and the production environment. At the same time, a series of improvements are made on the level of container for data information security problems in a production environment. The problem of container escape and container exhaustion of host resources is effectively prevented by container resource constraints of Cgroups and mandatory access control of SELinux. The container network security issues should be replied by prohibiting communication between containers that do not form a microservice, which can effectively reduce the risk that attackers misappropriate the information security of other containers or even host computers by holding a container. Finally, by uploading the fabricated image to the private cloud platform, the authentication mechanism is added to ensure the application of the image and its internal storage, as well as the safety of the data. The application of the accelerator control system can be deployed efficiently by the combination of Docker and EPICS technology, which can also save manpower and time. In this way, this technology has broad application prospects in the field of accelerator control.

REFERENCES

[1] Y. H. Guo *et al.*, "Progress of the Control Systems for the ADS injector II", in Proc. 15th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'15), Melbourne, Australia, Oct. 2015, pp. 709-712. doi:10.18429/JACoW-ICALEPCS2015-WEPGF011

[2] T. Birke, D. Faulbaum, M. G. Giacchini, D. Herrendörfer, P. Stange, and R. Lange, "BESSY Control System Adminis tration and Analysis Tools", in Proc. 11th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'07), Oak Ridge, TN, USA, Oct. 2007, paper RPPB28, pp. 671-673.

[3] CHEN Zhang-Nuo, "Development and design of upper application software for CAFe control system", 2020, University of Chinese Academy of Sciences (Institute of Modern Physics, Chinese Academy of Sciences) (In Chinese).

[4] Ranchal, Rohit, *et al.*, "Epics: A framework for enforcing s ecurity policies in composite web services," IEEE Transact ions on Services Computing 12.3 (2018): 415-428.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV031

[5] Zhang, Tong and Dylan Maxwell, "Cloud Computing Platform for High-level Physics Applications Development," presented at the 17th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper MOPHA167.

[6] LI Jiao-Sai, "Design of ECR ion source emittance measurement system", 2017, University of Chinese Academy of Sciences (Institute of Modern Physics, Chinese Academy of Sciences) (In Chinese).

[7] StreamDevice, https://paulscherrerinstitute.github.io/StreamDevice/

[8] EPICS IOC, https://epics.anl.gov/base/R3-14/12-docs/App Dev Guide/node4.html

[9] Qiao, Xianjie and Gang Li, "Research of streamdevice application based on EPICS", Nuclear Electronics and Detection Technology 31.10 (2011): 1073-1076.

[10] Anderson, Charles, "Docker [software engineering]", Ieee Software 32.3 (2015): 102-c3.

[11] Boettiger, Carl, "An introduction to Docker for reproducible research", ACM SIGOPS Operating Systems Review 49.1 (2015): 71-79.

[12] Bui, Thanh, "Analysis of docker security," arXiv preprint arXiv:1501.02967 (2015).

[13] REN Lan-Fang, Zhuang Xiao-Jun, FU Jun, "Research on Docker Container Security Protection Technology", Telecommunications Engineering Technology and Standardization (2020) (In Chinese).

[14] BI An Safety team. Docker Container Safety Analysis [EB\OL], https://www.frebuf.com/artcles/system/221319.html

[15] Docker compose YAML file, https://docs.docker.com/compose