Design and Implementation of High-availability PaaS Platform Based on Virtualization Platform

Zepeng Wen[1], Yan Liang[1] ,Gongliang Li*[2]
1.Institute of Computer Application, CAEP, Mianyang, China
2.College of Computer Science, Sichuan University, Chengdu, China
winzip@163.com, xuehuaiyan@163.com,ligl@icaep.com

***Abstract*—In view of the problems of the virtual platform failure migration mechanism under the cloud model and microservice architecture platform, the demand for redundant resources is large, and the failure recovery time is long. The key components such as the operating components, resource pools and middleware of the PaaS platform are analyzed, combined with virtualization The platform deployment application is practical. The PaaS platform high-availability architecture is designed to realize the PaaS platform high-availability deployment mode, which effectively improves the robustness of the PaaS platform and guarantees the business continuity of the application system running on the PaaS platform. This paper first analyzes the functional architecture and role positioning of the PaaS platform; secondly, it proposes high-availability design of operating components, resource pools and middleware, and at the same time maps the design to the physical deployment mode, the experimental results show that the design is feasible and effective.**

Keywords—Cloud model; Microservices; PaaS platform; High availability architecture; Robustness;

I. INTRODUCTION

With the widespread use of cloud models and microservice architectures in enterprise-level information systems, enterprise intranet virtualization platforms are not just pure information systems, but PaaS platforms that shield the underlying hardware and software features. Improve the efficiency of IT resources and applications and reduce operating costs through the PaaS platform. In the traditional server maintenance, if a server is down, the most is that the application on this server cannot provide services normally, which can be solved through cluster high availability and other methods; but in a cloud mode environment, if a virtual host If the end server is down, the virtual machines running on it will not work properly. Although virtualization manages all resources in a centralized manner, the impact area when a failure occurs is greatly increased, so the high-availability architecture of the PaaS platform is particularly important among virtualization platforms. It is not enough to simply protect the physical server, but also to protect important business continuity and enterprise data. While the virtualization platform provides flexibility, if a physical server containing multiple data storage virtual hosts fails, it will cause huge data loss.

At present, business continuity and data security are mainly guaranteed by the virtualization platform's own high-availability assurance mechanism. The virtualization platform ensures that a physical host in the resource pool is down after the physical host is formed into a resource pool and HA mechanism. Virtual hosts can be migrated to other physical hosts with rich resources in the resource pool. However, the high availability guarantee mechanism of the virtualization platform requires a lot of redundancy of resources, such as 25 virtual machines on a physical server. If the design index is to withstand 1 physical machine downtime and ensure business continuity, then one is required. Redundant physical server. After the enterprise information system is centralized, the container resource pool of the PaaS platform will be composed of hundreds of virtual hosts, so a large number of physical servers are required as backup resources for failure migration. At the same time, because microservices and containerized enterprise information systems are highly dependent on time synchronization and state consistency, the migration process of virtualization platforms is often time-consuming, making it difficult to recover virtual machines on downtime hosts in a short time, even if the service After recovery, it is difficult to guarantee the state consistency of the components in the PaaS platform.

In view of the high resource requirements of the virtualization platform's own high resource requirements and long failure recovery time, we analyzed the architecture of the PaaS platform and deployed components, resource pools, and middleware that involve business continuity and data storage. Available designs.

II. PAAS PLATFORM DEFINITION AND FUNCTIONS

A. PaaS platform definition

The PaaS platform integrates and packages the ability to use the cloud or other infrastructure and the ability to apply various technology middleware. Filter out technical details, provide a simple, consistent, and easy-to-use capability interface for application technology infrastructure to help the rapid construction of the data center in the foreground and business.

The PaaS platform provides a standardized platform for application R & D operation, covering from R & D, to release, to daily operation and maintenance, decoupling enterprise applications and computing resources, and carrying out R & D activities according to a unified process at low cost, reducing errors and improving efficiency.

B. PaaS platform features

The PaaS platform can provide developers with basic application services such as resource management, container

services, continuous integration, continuous delivery, and image warehouse. At the same time, it provides complete support for the application of the microservice architecture. Log management, operation analysis, middleware services and other functions help development and operation and maintenance personnel to reduce the burden in the product development iteration process.

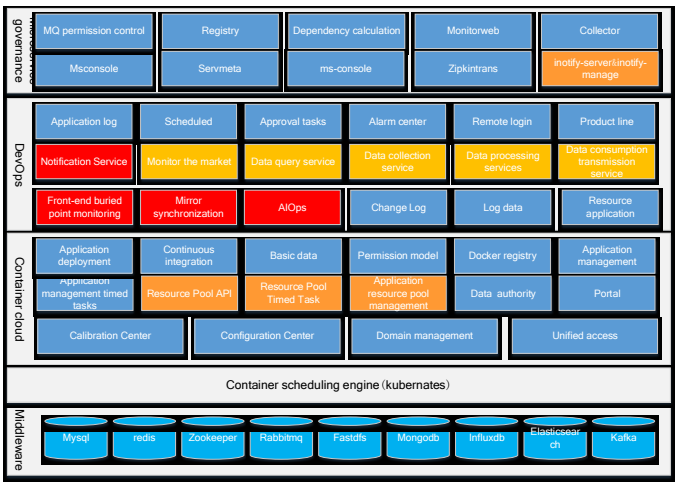

Fig. 1. PaaS platform functional architecture

III. PAAS PLATFORM HIGH AVAILABILITY DESIGN

A. Highly available architecture design

According to the operating components, middleware, data storage and deployment modes of the PaaS platform, the high-availability architecture design of the PaaS platform is composed of four parts, namely, high availability of running components, high availability of middleware, high availability of data storage, and high availability of deployment mode. The high-availability overall deployment architecture of the PaaS platform is shown in the following figure 2.

The PaaS platform running components are composed of open source components, framework components and resource pools that are physically deployed on the host, all of which are independent host deployments.

PaaS platform middleware is the middleware that supports the operation of PaaS platform, including database, file storage and message queue and other middleware.

PaaS platform data storage is the storage of important data files such as PaaS platform business application packages and pipeline-built image files.

The PaaS platform deployment mode refers to the above-mentioned virtual machines that require highly available components and resource pools, and need to be distributed to different physical machines to ensure true high availability.

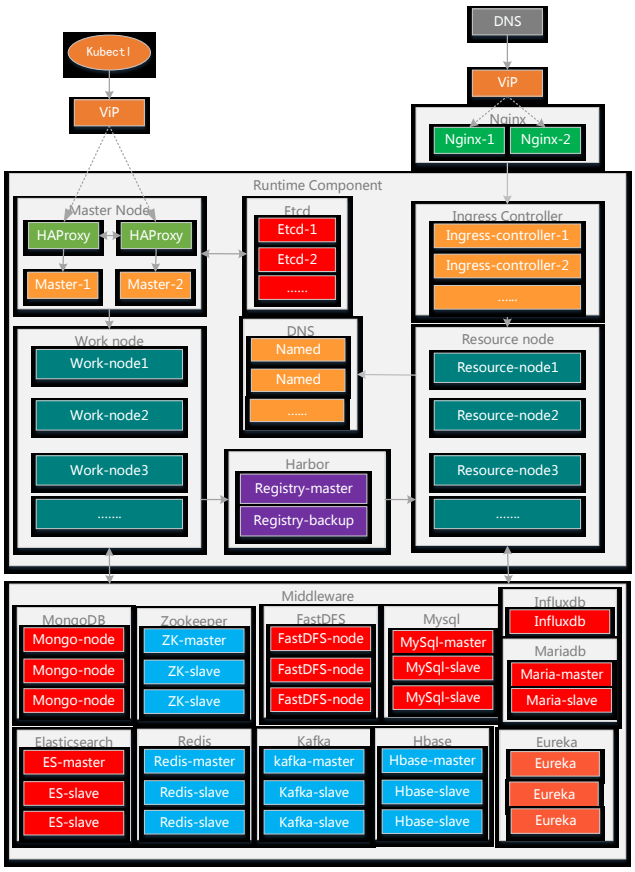

Fig. 2. PaaS high availability architecture

B. High availability design of running components

There are 7 types of components deployed on the host of the PaaS platform, namely Master Node, Etcd, Ingress, Word Node, harbor, DNS (named), and Resource Node. All components need to be designed for high availability. By sorting out the functions and technical architecture of the above components, the high-availability architecture of the PaaS platform is designed as shown in the figure below.

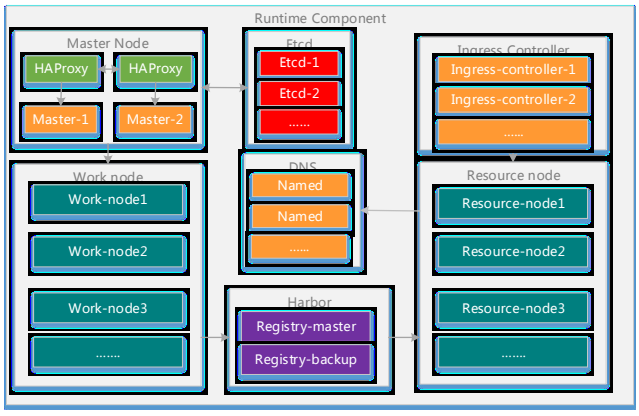

Fig. 3. PaaS platform components high availability architecture

The Master Node (Kubernates master node) is a scheduling and management node for the PaaS platform container. It is designed to use a dual node mode. The front end forms a

highly available cluster with HAProxy through VIP (Virtual IP).

Etcd is a storage database for PaaS platform network configuration and object status information. It is a core component of PaaS platform resource scheduling and operation. It is designed to use more than 3 nodes to form a highly available cluster. It is recommended to configure 5 nodes.

Ingress is a PaaS platform service discovery node. All business application resource pool access requests are forwarded by Ingress. The design uses more than 3 nodes to form a highly available cluster.

Word Node (Kubernates slave node) is a resource node for PaaS platform's own service operation. 120% resource redundancy is designed according to the service volume. It is recommended to use more than 3 hosts (8cpu 16GB).

DNS is a domain name server called by the internal business application of the PaaS platform. It cooperates with Ingress to complete the service discovery and request forwarding. All services called by the domain name need to be resolved by the DNS. The design uses 3 nodes to form a highly available cluster.

Resource Node (Resource Node) constitutes a node of the PaaS platform's resource pool, provides an operating environment for the business center or business applications, and designs 120% resource redundancy according to the business volume.

Harbor (mirror warehouse) is a warehouse for mirrors and basic mirrors created by the pipeline of the PaaS platform. It manages the business applications of the PaaS platform and running mirrors of historical versions. It is designed to use two nodes to form a highly available cluster.

C. *High availability design of middleware architecture*

The PaaS platform involves 10 types of middleware, namely Nginx, MongoDB, Zookeeper, Mariadb, FastDFS, MySQL, Influxdb, ElasticSearch, Redis, Kafka, Hbase and Eureka, as shown in the figure below.

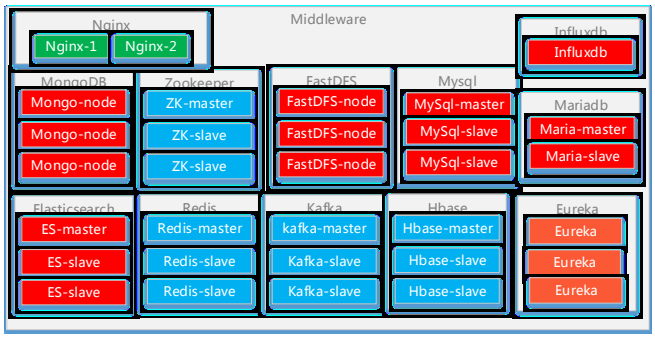

Fig. 4. PaaS platform middleware high availability architecture

Among them, Nginx, MongoDB, Mariadb, FastDFS, MySQL, ElasticSearch, Redis, Kafka are the key middleware for the normal operation of the PaaS platform, Influxdb is the monitoring and collection of platform microservices operation data, Hbase is the operation database of Pinpoint performance monitoring components, and Eureka is Microservices registry.

In order to ensure the stable and reliable operation of the PaaS platform, high-availability designs are made for key middleware, performance monitoring component middleware, and microservice registration centers. Influxdb is not designed for high-availability monitoring data collection middleware.

The middleware cluster design is mostly a master-two-slave cluster model, taking MySQL as an example. MySQL provides structured data storage services for the PaaS platform, including account information, configuration parameters, etc., to achieve a highly available cluster through a master and two slave methods. Among them, the master node provides write service, and the slave node provides read service. If the master node is down, the slave node will be automatically elected to undertake the write service of the master node, thereby achieving high availability of MySQL database service.

IV. HIGH AVAILABILITY DESIGN FOR DEPLOYMENT MODE

A. *Design Principles*

1) No more than 25 nodes on each host

There are no more than 25 nodes on each host, including Master Node, Etcd, Ingress, Word Node, harbor, DNS (named), Resource Node, or all types of nodes.

2) The principle of resource redundancy

Configure host resources according to 120% resource redundancy.

$$num_{node} = \max\left(\left\lceil r_{cpu} \times 120\% \div p_{cpu} \right\rceil, \left\lceil r_{ram} \times 120\% \div p_{ram} \right\rceil\right)$$

For example, the resource node needs 16cpu, 40GB memory, according to 120% resource redundancy, need to configure 19.2cpu, 48GB memory, such as 8cpu 16GB host, you need to configure 3 hosts.

3) Master node distribution principle

The number of Master nodes is consistent with the number of Master hosts, and each host is placed with a node.

$$num_{worknode} \%2 = 1, num_{worknode} \geq 3$$

$$num_{computer} = \left\lceil \frac{1}{2} num_{worknode} \right\rceil$$

Work Node is generally an odd number with a fixed number greater than or equal to 3, and no more than 2 work nodes on each host. If the number of nodes is less than 5, it will be distributed on two hosts, if it is greater than or equal to 5, it will be distributed on 3 hosts.

5) Resource node distribution principle

$$num_{node} = \min\left(num_{resourcenode} \times \frac{num_{computer}}{num_{computer} - 1} \times \frac{1}{num_{computer}}, 20 \right)$$

ccording to the host downtime, one can still meet the normal operation of 100% of the resource nodes. If the number of nodes requires 100, 5 hosts can be used. After redundancy, the number of nodes is $100 \times \frac{6}{6-1} = 120$, $\min\left(120 \times \frac{1}{6}, 20\right) = 20$. then 20 nodes are allocated on each host .

6) Distribution principle of middleware cluster

The middleware cluster of the PaaS platform is basically 3 nodes, and all middleware cluster nodes are distributed to 3 host machines.

B. Highly available architecture design

The PaaS platform running component provides operation support services for the PaaS platform's resource scheduling management. When the component is highly available, multi-node high availability clusters are designed. In order to avoid the high availability failure caused by the downtime of the host of the high availability cluster host, all hosts are distributed to different physical machines during deployment, as shown in the following figure.

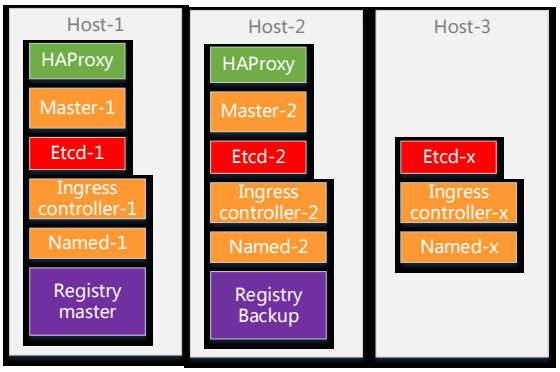

Fig. 5. Highly available design of component deployment mode

Cluster nodes of the same type are distributed on different host machines. The number of host machines is consistent with the number of master nodes. If the number of cluster nodes of other components is greater than the number of master nodes, all cluster nodes are relatively evenly distributed on the host machine.

C. High availability design of resource pool deployment mode

PaaS platform resource pools include Work Node and Resource Node, where Work Node is a resource pool for PaaS platform resource management applications, and Resource Node is a resource pool for business middle offices and business applications. Work Node is generally a fixed number. If the number of nodes is less than 5, it will be distributed on two host machines. If it is greater than or equal to 5, it will be distributed on three host machines. The deployment mode is shown in the figure below.

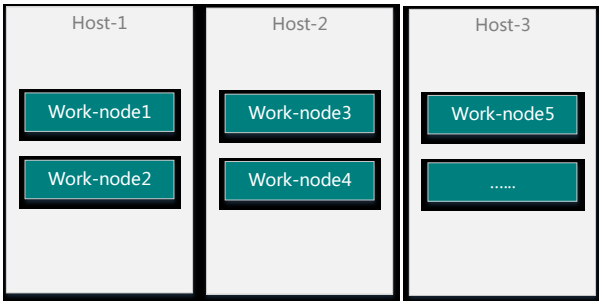

Fig. 6. Work Node deployment mode high availability design

The number of Resource Nodes is directly proportional to the scale of business applications, and the number is generally large; the number of resource nodes is 120% of the actual demand of the business. If the number of nodes / 4 <20, they are evenly distributed on the four host machines. If the number of nodes / 4 > = 20 is based on 20 units and distributed to a corresponding number of hosts. The deployment mode is shown in the figure below.

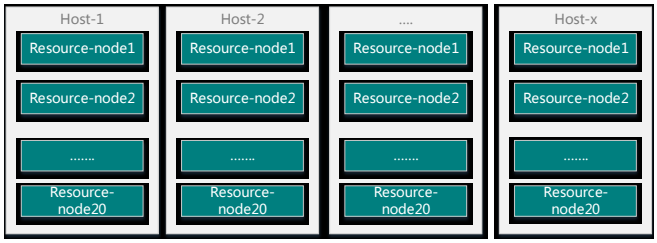

Fig. 7. Resource Node deployment model high availability design

D. High availability design for middleware deployment mode

The middleware cluster of the PaaS platform is basically three nodes. All middleware cluster nodes are distributed to three host machines.

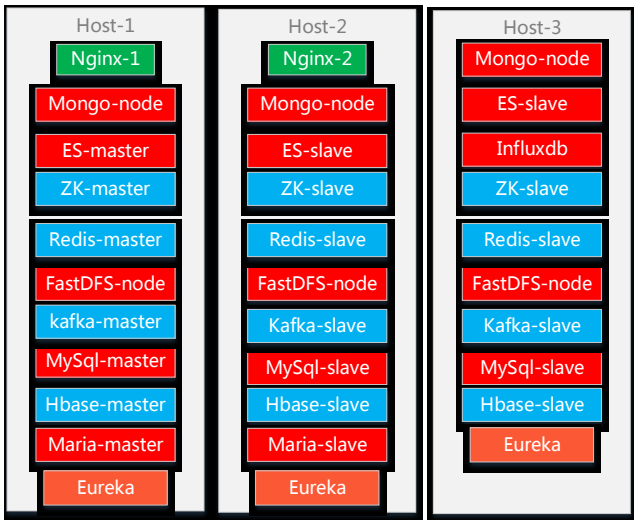

Fig. 8. High availability design for middleware deployment mode

V. VERIFICATION AND RESULT ANALYSIS

A. Experimental verification environment

In the experimental environment, the above high-availability design was experimentally verified, and through 4 physical servers, the PaaS platform high-availability design architecture was verified. The physical host configuration and the number of virtual machines are shown in the following table.

TABLE I. HOST CONFIGURATION LIST

	Host	CPU	Mem	Storage	VM Num
1	Host1	Intel Xeon® Silver 4110 CPU @2.10GHz;	128	8 SAS 10K 1.2TB RAID5	13
2	Host2	Intel Xeon® Silver	128	6 SAS 10K	12

		4110 CPU @2.10GHz;		1.2TB RAID5	
3	Host3	Intel Xeon® Silver 4110 CPU @2.10GHz;	128	6 SAS 10K 1.2TB RAID5	14
4	Host4	Intel Xeon® Silver 4110 CPU @2.10GHz;	128	6 SAS 10K 1.2TB RAID5	12

TABLE II. PAAS PLATFORM DEPLOYMENT MANIFEST

	category	Host1	Host2	Host3	Host4
1	Components	HAProxy Master-1 Etcd-1 Ingress controller-1 Named-1 Registry master	HAProxy Master-2 Etcd-2 Ingress controller-2 Named-2 Registry backup	Etcd-3 Ingress controller-3 Named-3	Etcd-4 Ingress controller-4 Named-4
2	Middleware		Mongo-node Influxdb ZK-slave Redis-slave FastDFS-node Kafka-slave Mysql-slave Hbase-slave Maria-slave	Nginx-1 Mongo-node ES-slave ZK-slave Redis-slave FastDFS-node Kafka-slave Mysql-slave Hbase-slave Maria-slave Eureka	Nginx-2 Mongo-node ES-master ZK-master Redis-master FastDFS-node Kafka-master Mysql-master Hbase-master Maria-master Eureka
3	Work Node	Work-node Work-node ……	Work-node Work-node ……	Work-node Work-node ……	Work-node Work-node ……
4	Resource node	Resource-node Resource-node ……	Resource-node Resource-node ……	Resource-node Resource-node ……	Resource-node Resource-node ……

B. Test methods and results

We tested the two extreme scenarios of power failure and network disconnection that caused physical server downtime to verify whether the high-availability design can maintain business continuity.

Power-off scenario: Turn off the power of one of the four physical servers directly.

Network disconnection scenario: Directly disconnect the network of one of the four physical servers.

TABLE III. HIGH AVAILABILITY TEST RESULT RECORD

	Test items	Test Results
1	Physical host 1 is disconnected from the network, other hosts are operating normally	Keep up and running
2	Physical host 2 is disconnected from the network, other hosts are operating normally	Keep up and running
3	Physical host 3 is disconnected from the network, other hosts are operating normally	Keep up and running
4	Physical host 4 is disconnected from the network, other hosts are operating normally	Keep up and running
5	Physical host 1 is powered off, other hosts are operating normally	Keep up and running
6	Physical host 2 is powered off, other hosts are operating normally	Keep up and running
7	Physical host 3 is powered off, other hosts are operating normally	Keep up and running
8	Physical host 4 is powered off, other hosts are operating normally	Keep up and running

Through the analysis of the test results, this paper fully meets the design specifications for the high-availability architecture design and deployment model design of the PaaS platform. When a single physical service is down due to extreme conditions, it can ensure business continuity and normal operation of data services.

VI. CONCLUSION

This article focuses on analyzing the difficulties faced by the current virtualization platform failure migration and high availability in the cloud model and microservice architecture platform. On this basis, the architecture of the PaaS platform is designed for high availability, and the high-availability deployment model of the PaaS platform in the virtualization platform is also designed. By distributing the highly available nodes on different physical hosts, the service unavailability caused by the physical host downtime is avoided, and business continuity is guaranteed. Experimental verification proves the effectiveness of the design. However, when the business system is fully loaded and the resource pool host load is high, the automatic service migration may fail. The design of redundant resource pools needs to be added in the future. Improve the robustness of the PaaS platform.

REFERENCES

[1] Xinhui Yuan, Yong Liu, Fengbin Qi, Cloud framework for hierarchical batch-factor algorithm, Journal of computer Applications, 2014, 34(3): 690-694.

[2] Yao Ke, Baohua Zhao, Yugui Qu, Software Reliability Analysis of Component Based System, Journal of Beijing University of Posts and Telecommunications.2005,28(6):115-119.

[3] Zhang Qi, Cheng Lu, Boutaba Raouf. Cloud computing: state-of-the-art and research challenges. Journal of Internet Services and Applications, 2010, 1(1): 7-18.

[4] JaegerT.,SehiffmanJ.Outlook: Cloudy with a Chanee of Seeurity Challenges and ImprovementS[J].Seeurity&Privaey,IEEE,2010,8(1).

[5] Mell P, Grance T. The NIST definition of cloud computing (draft). NIST Special Publication, 2011, 800(145): 7.

[6] Deqin Zhou, Hai Jin, Gang Chen, Research on fault tolerant mechanism of computing for disaster tolerant cloud, Communications of the China Computer Federation,2012, 8 (7): 58-64.

[7] Schroeder B, Gibson G A. A large-scale study of failures in high-performance computing systems. IEEE Transactions on Dependable and Secure Computing, 2010,7(4): 337-350.

[8] Bessani A, Correia M, Quaresma B, et al. DepSky: dependable and secure storage in a cloud-of-clouds. ACM Transactions on Storage (TOS), 2013, 9(4): 12-12.

[9] Buyya,Rajkumar.Cloud computing and emerging IT platforms:Vision, hype, and reality for delivering computing as the 5th utility[J].Future Generation Computer Systems,2009,25(6):599-611.