



(51) International Patent Classification:
G06F 9/50 (2006.01)

(21) International Application Number:
PCT/CN2015/084851

(22) International Filing Date:
22 July 2015 (22.07.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
201410359705.2 25 July 2014 (25.07.2014) CN

(71) Applicant: **HANGZHOU H3C TECHNOLOGIES CO., LTD.** [CN/CN]; 466 Changhe Road, Binjiang District, Hangzhou, Zhejiang 310052 (CN).

(72) Inventors: **LV, Zhenfeng**; Room 730, Oriental Electronic BLD., No.2, Chuangye Road, Shangdi Information Industry Base, Haidian District, Beijing 100085 (CN). **SUN, Songer**; Room 730, Oriental Electronic BLD., No.2, Chuangye Road, Shangdi Information Industry Base, Haidian District, Beijing 100085 (CN).

(74) Agent: **BEIJING BESTIPR INTELLECTUAL PROPERTY LAW CORPORATION**; Room 1209, B Block of Jiahua Building, No.9 Shangdi 3rd Street, Haidian District, Beijing 100085 (CN).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

- with international search report (Art. 21(3))
- before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))

(54) Title: SCHEDULING OF SERVICE RESOURCE

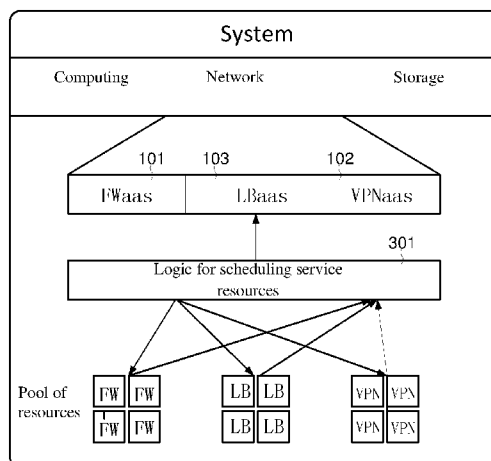


Fig.3

(57) Abstract: An OpenStack controller periodically detects performance parameters of multiple resource nodes and determines allocable resource nodes among the resource nodes according to the detected performance parameters. The controller allocates, from the allocable resource nodes, an optimum resource node for a user according to a preset scheduling policy upon reception of a resource acquiring request of the user, and returns the optimum resource node to the user.

SCHEDULING OF SERVICE RESOURCE

Field**5 Background**

[0001] The OpenStack is an emerging open source software architecture to provide a cloud computing management solution, which intends to build an open and extensible framework with which various resources in a cloud environment are managed. Due to its open-source and open accessibility, the OpenStack architecture is supported by the majority of predominant
10 manufacturers, and has become a widely applied open-source cloud computing solution.

Summary**Brief Description of the Drawings**

15 **[0002]** Fig.1 illustrates a schematic diagram of a hardware structure of a system in an example;
[0003] Fig.2 illustrates a framework diagram of a controller in an example;
[0004] Fig.3 illustrates a framework diagram of another system in an example; and
[0005] Fig.4 illustrates a flow chart of a method of scheduling service resources in an
20 example.

Detailed Description of the Embodiments

[0006] The present disclosure relates to a cloud computing system which enables computing, storage and other resources to be provided to users over the Internet. In some cases a cloud
25 computing system may be used by a plurality of users, and resources may be managed and dynamically allocated from one user to the other by the system controller according to the demand for resources at any one time. The following description refers to the OpenStack system, but the principles disclosed herein may also be applied to other cloud computing systems and controllers. A modular architecture provided by the OpenStack typically includes various

modular resources in a cloud environment, such as computing, storage and network, to provide an integral framework solution. In the cloud environment, a cloud computing data center typically provides a user with a virtual host lease service, and since there is no physical boundary for the virtual host lease service, it is difficult for the user herself or himself to deploy a physical network security device. Hence, the cloud computing data center needs to provide the user with network security solutions such as Firewall as a Service (FWaaS).

[0007] For example, the user can operate directly on a user interface provided by the OpenStack to purchase a virtual firewall, and after the user purchases the virtual firewall, an OpenStack console may be connected automatically with a physical firewall to create the virtual firewall for the user and to configure the virtual firewall correspondingly. After the virtual firewall is configured, the user can perform service management on her or his own virtual firewall.

[0008] Reference can be made to Fig.1, which illustrates an OpenStack system framework in an example of this disclosure. In the OpenStack system framework, service modules such as an FWaaS 101, a Virtual Private Network as a Service (VPNaaS) 102, and a Load Balance as a Service (LBaaS) 103 are preliminarily defined at present, and these service modules can be controlled centrally to provide the user with a relative complete network security solution.

[0009] As illustrated in Fig.1, various security services, e.g., Firewall, LB, VPN, etc., are all managed and distributed centrally by the OpenStack controller as service resources in the OpenStack framework. But if there is a pool of service resources formed of multiple resource nodes, no solution is provided in the OpenStack framework to select and allocate an optimum resource from the pool of resources. Thus, the resources may be allocated inappropriately and consequently underutilized; for example, there may be some resource node being overloaded while other resource nodes stay idled.

[0010] In an example of the disclosure, in addition to the above OpenStack framework, a logic for scheduling the service resources, which operates on the OpenStack controller, determines allocable resource nodes among the respective resource nodes by periodically detecting performance parameters of the respective resource nodes, and allocates an optimum resource node from the allocable resource nodes for the user according to a preset scheduling

policy, thereby optimizing the allocation of the service resources in the OpenStack framework and scheduling the resources flexibly and dynamically, so as to improve the utilization ratio of the resources.

[0011] The OpenStack controller may be embodied in a hardware structure as illustrated in Fig.2. As shown in Fig.2, the OpenStack controller 20 may include a processor (CPU) 210, a machine-readable storage medium 220, and a network interface 230, all of which are connected with each other through an internal bus 240.

[0012] The machine-readable storage medium 220 is configured to store instruction codes. When the instruction codes are executed by the CPU 210, operations generally including a function of scheduling service resources are performed.

[0013] The CPU 210 communicates with the machine-readable storage medium 220 to read and execute the instruction codes stored in the machine-readable storage medium 220 to achieve the function of scheduling the service resources.

[0014] The machine-readable storage medium 220 may be any electronic, magnetic, optic or other physical storage devices, which can include or store information such as executable instructions, data, etc. For example, the machine-readable storage medium 220 may be a Random Access Memory (RAM), a volatile memory, a nonvolatile memory, a flash memory, a storage driver (e.g., a hard disk driver), a solid-state hard disk, any type of storage disk (e.g., an optic disk, a DVD, etc.) or the like, or any combination thereof. Moreover, any machine-readable storage medium described in the disclosure can be non-transitory.

[0015] Reference can be made to Fig.3, which illustrates an OpenStack framework in another example of the disclosure. In addition to the OpenStack framework illustrated in Fig.1, the OpenStack framework illustrated in the example includes a logic for scheduling service resources (abbreviated as a scheduling logic hereinafter) 301, which is arranged in the OpenStack controller and is for receiving an upper Application Programming Interface (API) call and allocating the most appropriate resource node for the user flexibly and dynamically according to a preset scheduling policy.

[0016] Particularly, the CPU of the OpenStack controller reads and executes the logic for scheduling service resources 301 in the machine-readable storage medium in an operational

flow as illustrated in Fig.4.

[0017] Block 410 is for periodically detecting performance parameters of respective resource nodes.

[0018] Block 420 is for determining allocable resource nodes among the respective resource nodes according to the detected performance parameters.

[0019] Block 430 is for allocating an optimum resource node from the allocable resource nodes for a user according to a preset scheduling policy upon reception of a resource acquiring request of the user, and to return the optimum resource node to the user.

[0020] In the example, the OpenStack controller executing the scheduling logic can receive the resource acquiring request from the user. In an implementation, the resource acquiring request may be a resource acquiring request message sent by the user operating on a user interface provided by the OpenStack. The resource acquiring request message may include a service creation request and even a performance requirement set by the user for a service. For example, the user can operate on the user interface provided by the OpenStack to send a resource acquiring request message, to request the OpenStack controller for a firewall service with a required processing capacity of 1Gbbp/s.

[0021] The OpenStack controller executing the scheduling logic can create a corresponding service and allocate an appropriate resource node for the user upon reception of the resource acquiring request message sent by the user operating on the user interface provided by the OpenStack, and returns a determination result to the user when an optimum resource node for the user is determined by the OpenStack controller.

[0022] In the example, the OpenStack controller executing the scheduling logic can periodically detect different performance parameters of the respective resource nodes in the pool of resources, to monitor in real time loads and operating states of the respective resource nodes in the pool of resources. For example, the OpenStack framework typically includes service modules adapted to provide the user with a network security solution, such as, the FWaaS, the VPNaaS, the LBaaS, etc., so the OpenStack controller executing the scheduling logic can periodically detect the different performance parameters of the respective resource nodes in the pool of resources, thereby monitoring in real time the loads and operating states of

the respective resource nodes in the pool of resources.

[0023] Particularly, when the OpenStack controller executing the scheduling logic periodically detects the different performance parameters of the respective resource nodes in the pool of resources, the performance parameters may be preconfigured performance parameters, or may be performance parameters determined according to a particular performance requirement, which is required by the user for the service resource and is carried in the resource acquiring request message, which is not be specifically limited in the embodiment.

[0024] For example, in an example of this disclosure, the performance parameters may include CPU utilization ratios and memory utilization ratios of the resource nodes, and other service performance parameters which can characterize resource availability of the resource nodes; and the service performance parameters may include parameters such as service traffic, service response time, etc., where a resource node with a smaller service traffic and a resource node with shorter service response time have higher resource availability.

[0025] In the example, the OpenStack controller executing the scheduling logic may determine the allocable resource nodes among the respective resource nodes according to a result of periodically detecting the different performance parameters of the respective resource nodes in the pool of resources, and allocates an appropriate resource node from the allocable resource nodes for the user according to a preset resource scheduling policy after determining the allocable resource nodes among the respective resource nodes.

[0026] In the example, the OpenStack controller executing the scheduling logic may allocate of the appropriate resource node from the allocable resource nodes for the user according to the preset resource scheduling policy, by comparing at least one of the CPU utilization ratios and the memory utilization ratios of the respective resource nodes with a threshold and categorizing the respective resource nodes into allocable resource nodes and non-allocable resource nodes.

For example, the OpenStack controller executing the scheduling logic may firstly determine whether at least one of the CPU utilization ratio and the memory utilization ratio of each resource node is above the preset threshold; if at least one of the CPU utilization ratio and the memory utilization ratio of one resource node is above the preset threshold, the OpenStack controller categorizes the resource node as a non-allocable resource node; otherwise, if at least

one of the CPU utilization ratio and the memory utilization ratio of one resource node is not above the preset threshold, the OpenStack controller categorizes the resource node as an allocable resource node.

[0027] The preset threshold is not specifically limited in the discourse and may be preset according to actual needs of the user. For example, if the user intends to request a firewall service with a processing capacity of 1Gbps, and the processing capacity of a resource node can not satisfy the requirement of 1Gbps when the CPU utilization ratio of the resource node is above 70%, the threshold can be set to 70%.

[0028] In the example, the non-allocable resource nodes may not be allocated; and upon reception of the resource acquiring request message from the user, the OpenStack controller executing the scheduling logic dynamically and flexibly allocates the allocable resource nodes for the user according to the priorities and the service performance parameters of the respective allocable resource nodes, or other performance parameters influencing directly the service.

[0029] The OpenStack controller executing the scheduling logic may allocate the allocable resource nodes by firstly comparing the priorities of the respective resource nodes and then allocating resource node having a highest priority for the user as an optimum resource node. If there are multiple resource nodes having the highest priority among the allocable resource nodes, the service performance parameters of the multiple resource nodes having the highest priority may be further compared and respective resource availabilities of the multiple resource nodes having the highest priority can be acquired, to determine an optimum resource node to be allocated for the user.

[0030] For example, in an example of the disclosure, if there are multiple resource nodes having the highest priority among the allocable resource nodes, the OpenStack controller executing the scheduling logic may further compare the service traffics of the multiple resource nodes having the highest priority, selects a resource node having a lowest service traffic from the multiple resource nodes having the highest priority, and allocates the selected resource node for the user.

[0031] If the multiple resource nodes having the highest priority have the same service traffic, the OpenStack controller executing the scheduling logic may further compare the service

response time of the multiple resource nodes having the highest priority, selects a resource node having shortest response time from the resource nodes having the highest priority and the same service traffic, and allocates the selected node for the user.

[0032] If the multiple resource nodes having the highest priority still have the same service response time, the OpenStack controller executing the scheduling logic may allocate randomly one resource node from the resource nodes for the user, or may further compare other performance parameters which can characterize the availabilities of the service resources until an appropriate resource node is selected and allocated for the user.

[0033] In the above examples, the OpenStack controller executing the scheduling logic monitors in real time the performance parameters of the respective resource nodes, and allocates the appropriate resource node for the user according to the preset scheduling policy, thereby optimizing the allocation of the service resources in the OpenStack architecture and scheduling flexibly and dynamically the resources, so as to improve the utilization ratio of the resources without causing a situation that some resource node is overloaded while other resource nodes stay idled.

[0034] In another example of the disclosure, there is further illustrated a particular application.

[0035] Reference is still made to Fig.3. If there are four firewall resource nodes, denoted as A, B, C and D respectively, in a current pool of resources in the OpenStack architecture illustrated in Fig.3, a user requests for a firewall service with a processing capacity of 1Gbbp/s by inputting the following message:

Firewalls Request (requesting for a firewall service):

GET /v2.0/fw/firewalls.json

User-Agent: python-neutronclient

Accept: application/json

[0036] After the OpenStack controller executing the above-mentioned scheduling logic obtains the input message, the OpenStack controller may allocate an appropriate resource node for the user according to a preset scheduling policy. The OpenStack controller may allocate the

appropriate resource node for the user by monitoring in real time performance parameters such as CPU utilization ratios, memory utilization ratios, service traffics, service response time, etc., of respective firewall resource nodes, determining allocable resource nodes according to the monitoring result, and then allocating an appropriate resource node, from the allocable resource nodes, for the user.

[0037] If both the CPU utilization ratios and the memory utilization ratios are above 70%, the processing capacities of the respective firewall resource nodes can not satisfy the requirement of 1Gbps; hence, the respective firewall resource nodes can be categorized into allocable firewall resource nodes and non-allocable firewall resource nodes by using a threshold of 70%.

[0038] If both the CPU utilization ratio and the memory utilization ratio of the firewall resource node A in the pool of resources are above 70%, and both the CPU utilization ratios and the memory utilization ratios of the firewall resource nodes B, C and D are below 70%, the firewall resource node A is a non-allocable firewall resource node, and the firewall resource nodes B, C and D are allocable firewall resource nodes.

[0039] After the firewall resource nodes are categorized, the OpenStack controller allocates an optimum firewall resource node, from the allocable firewall resource nodes B, C and D, for the user. The OpenStack controller may allocate the optimum firewall resource node by firstly comparing priorities of the firewall resource nodes B, C and D, and selecting and allocating a firewall resource node having the highest priority for the user.

[0040] If the priorities of the firewall resource nodes B, C and D are 1, 2 and 2 respectively, the OpenStack controller may allocate the firewall resource node B for the user. Here, a smaller value of the priority indicates a higher priority.

[0041] If the priorities of the firewall resource nodes B, C and D are 1, 1 and 2 respectively, then the service traffics of the resource nodes B and C are further compared; and if there the resource node B has a lower service traffic than the resource node C, the OpenStack controller may allocate the resource node B for the user.

[0042] If the resource nodes B and C have the same service traffic, a further comparison is performed on the service response time of the resource nodes B and C; and if the resource node B has shorter service response time than the resource node C, then the OpenStack controller

may allocate the resource node B for the user.

[0043] If the resource nodes B and C still have the same service response time, the OpenStack controller may allocate randomly one resource node from the resource nodes B and C for the user or may further compare another performance parameter which can characterize the availabilities of the service resources until an appropriate resource node is selected and allocated for the user.

[0044] The OpenStack controller may return information about the optimum resource node to the user when determining the optimum resource node for the user.

[0045] For example, if the OpenStack controller finally allocates the resource node B for the user, returned information about the firewall service may be reflected as follows:

Firewalls: Response

"firewalls":

"admin_state_up": true,

"description": "**",

"firewall_policy_id": "c69933c1-b472-44f9-8226-30dc4ffd454c",

"id": "3b0ef8f4-82c7-44d4-a4fb-6177f9a21977",

"name": "B",

"status": "ACTIVE",

"tenant_id": "45977fa2dbd7482098dd68d0d8970117".

[0046] The returned information about the firewall service includes a service ID, a firewall node ID, a firewall state, a firewall description, a user ID, and other information.

[0047] In the examples above, the OpenStack controller executing the scheduling logic monitors in real time the performance parameters of the respective resource nodes, and allocates the appropriate resource node for the user according to the preset scheduling policy, thereby optimizing the allocation of the service resources in the OpenStack architecture and scheduling the resource flexibly and dynamically, so as to improve the utilization ratio of the resources

without causing a situation where some resource node is overloaded while other resource nodes stay idled.

CLAIMS

1. A method for scheduling service resources, applied to a controller, wherein the method comprises:

5 periodically detecting performance parameters of a plurality of resource nodes;
determining allocable resource nodes among the resource nodes according to the detected performance parameters; and
allocating, from the allocable resource nodes, an optimum resource node for a user according to a preset scheduling policy upon reception of a resource acquiring request of the
10 user, and returning the optimum resource node to the user.

2. The method according to claim 1, wherein the performance parameters include parameters selected from the list comprising: CPU utilization ratio, memory utilization ratio, service traffic, service response time and current resource availability.

3. The method according to claim 2, wherein said determining allocable resource nodes among
15 the resource nodes according to the detected performance parameters comprises:

determining whether the CPU utilization ratio or the memory utilization ratio of each resource node is above a preset threshold; and

determining that a resource node is a non-allocable resource node if it is determined that the CPU utilization ratio or the memory utilization ratio of the resource node is above the preset
20 threshold; otherwise, determining the resource node as an allocable resource node.

4. The method according to claim 2, wherein said allocating, from the allocable resource nodes, an optimum resource node for a user according to a preset scheduling policy comprises:

comparing priorities of the allocable resource nodes; and

allocating an allocable resource node having a highest priority for the user as the optimum
25 resource node.

5. The method according to claim 4, wherein said allocating, from the allocable resource nodes,

an optimum resource node for a user according to a preset scheduling policy comprises:

if there are a plurality of allocable resource nodes having the same highest priority, comparing the service performance parameters of the plurality of allocable resource nodes having the highest priority; and

5 selecting an allocable resource node having a highest current resource availability, from the plurality of allocable resource nodes having the highest priority, as the optimum resource node, and allocating the optimum resource node for the user.

6. The method according to claim 2, wherein, for any one of the resource nodes, the lower the service traffic, the higher the resource availability; and the shorter the service response time, the
10 higher the resource availability.

7. The method according to claim 1, wherein the controller is a Openstack controller.

8. A machine-readable storage medium, storing thereon machine readable instructions, which are executable by a processor to:

periodically detect performance parameters of a plurality of resource nodes;

15 determine allocable resource nodes among the resource nodes according to the detected performance parameters; and

allocate, from the allocable resource nodes, an optimum resource node for a user according to a preset scheduling policy upon reception of a resource acquiring request of the user, and return the optimum resource node to the user.

20 9. The machine-readable storage medium according to claim 8, wherein the performance parameters include parameters selected from the list comprising: CPU utilization ratio, memory utilization ratio, service traffic, service response time and current resource availability.

10. The machine-readable storage medium according to claim 9 the instructions to determine allocable resource nodes among the resource nodes according to the detected performance
25 parameters include instructions to:

determine whether at least one of the CPU utilization ratio and the memory utilization ratio

of each resource node is above a preset threshold; and

determine one resource node as a non-allocable resource node if it is determined that at least one of the CPU utilization ratio and the memory utilization ratio of the resource node is above the preset threshold; otherwise, determining the resource node as an allocable resource node.

11. The machine-readable storage medium according to claim 9, wherein the instructions to allocate, from the allocable resource nodes, an optimum resource node for a user according to a preset scheduling policy include instructions to:

compare priorities of the respective allocable resource nodes; and

allocate an allocable resource node having a highest priority for the user as the optimum resource node.

12. The machine-readable storage medium according to claim 11, wherein the instructions to allocate, from the allocable resource nodes, an optimum resource node for a user according to a preset scheduling policy include instructions to:

if there are a plurality of allocable resource nodes having the same highest priority, compare the service performance parameters of the plurality of allocable resource nodes having the highest priority; and

select an allocable resource node having a highest current resource availability, from the plurality of allocable resource nodes having the highest priority, as the optimum resource node, and allocate the optimum resource node for the user.

13. The machine-readable storage medium according to claim 9, wherein, for any one of the resource nodes, the lower the service traffic, the higher the resource availability; and the shorter the service response time, the higher the resource availability.

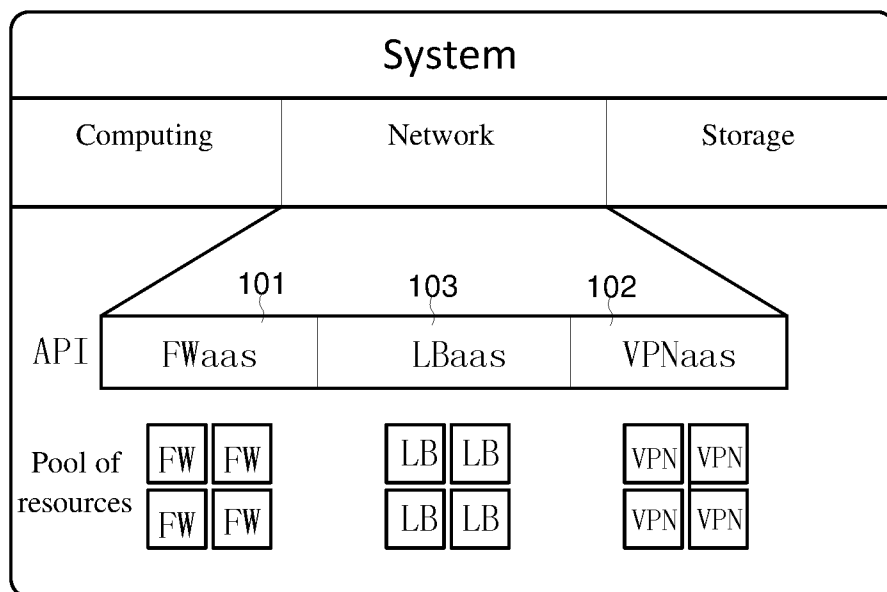


Fig.1

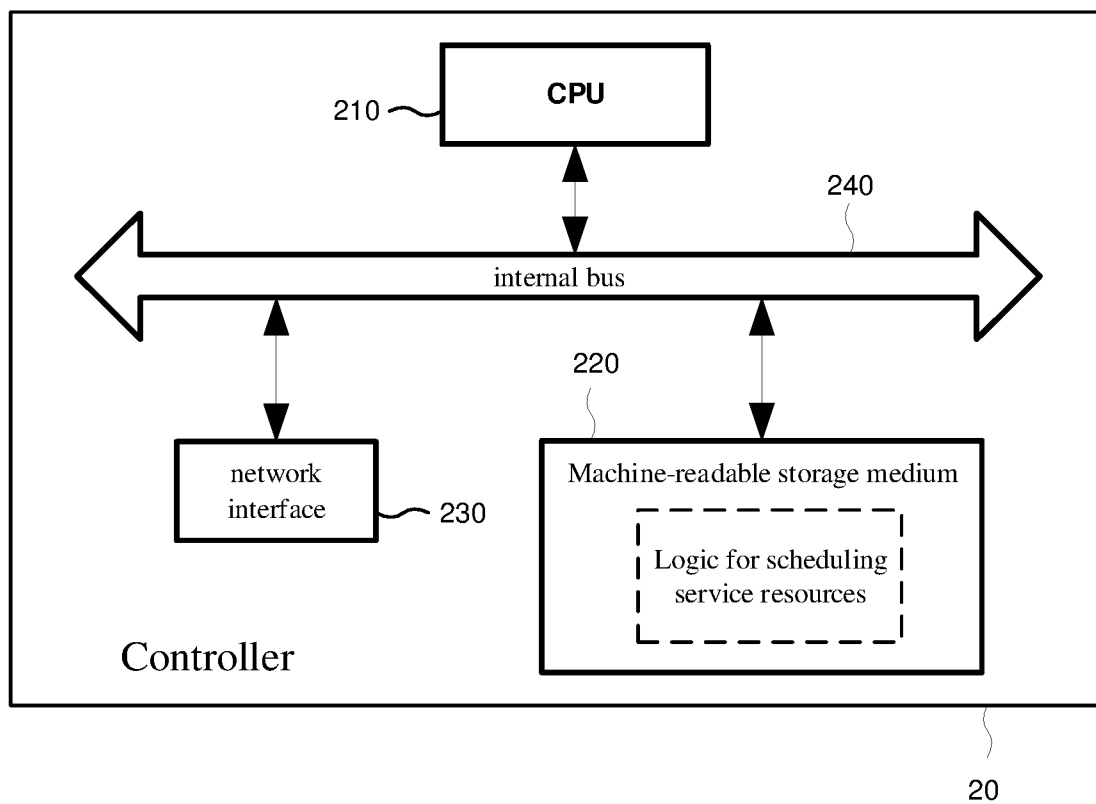


Fig.2

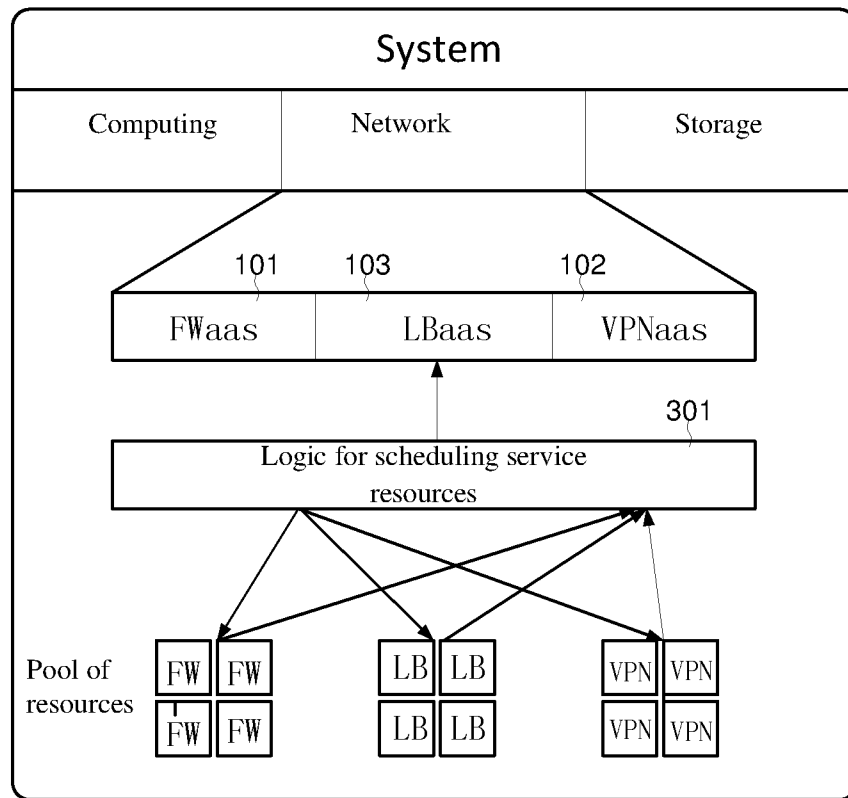


Fig.3

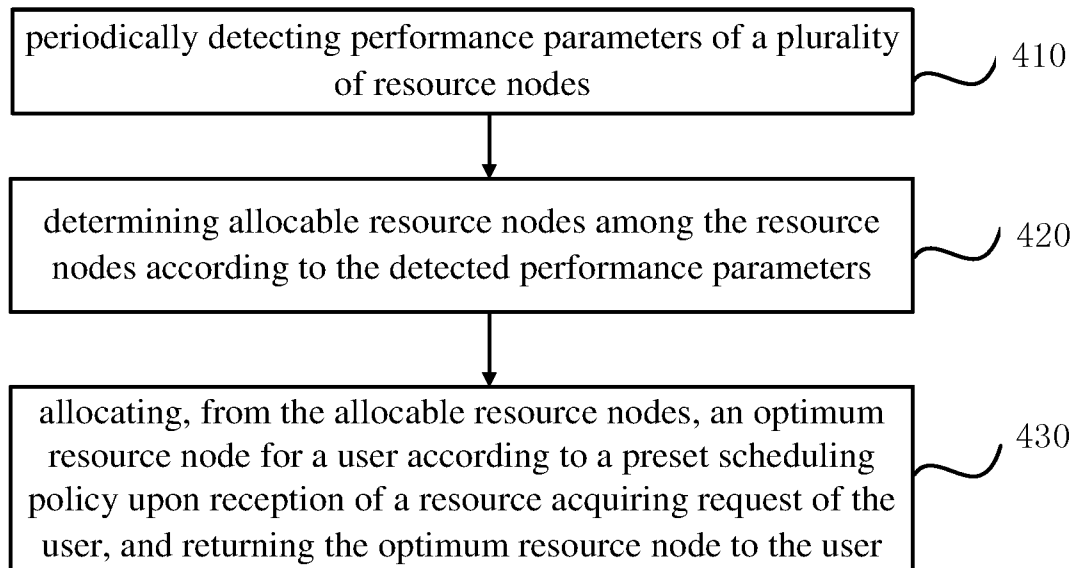


Fig.4

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2015/084851

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/50(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 9

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

ven,cnabs,entxt,entxt,cnki:cloud, computing, OpenStack, schedul+, policy, strategy, performance, parameter?,attribute?, resource, node, utilization, utility, priority, allocat+

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WANG, Nan. "The dynamic scheduling and management of computing resource in cloud platform based on OpenStack" <i>China's Outstanding Master's Degree dissertation full-text database</i> , No. No. 9, 15 September 2013 (2013-09-15), pages 13-14, 21-36, 42	1-3, 6-10, 13
Y	WANG, Nan. "The dynamic scheduling and management of computing resource in cloud platform based on OpenStack" <i>China's Outstanding Master's Degree dissertation full-text database</i> , No. No. 9, 15 September 2013 (2013-09-15), pages 13-14, 21-36, 42	4-5, 11-12
Y	CN 103576827 A (TIAN, WENHONG) 12 February 2014 (2014-02-12) abstract, paragraph [0045]	4-5, 11-12
Y	US 2014108639 A1 (IBM) 17 April 2014 (2014-04-17) paragraphs [0064]-[0065], figure 4	5, 12
A	CN 103338241 A (HEFEI UNIVERSITY OF TECHNOLOGY) 02 October 2013 (2013-10-02) the whole document	1-13



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

12 October 2015

Date of mailing of the international search report

24 November 2015

Name and mailing address of the ISA/CN

STATE INTELLECTUAL PROPERTY OFFICE OF THE
P.R.CHINA
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing
100088, China

Facsimile No. (86-10)62019451

Authorized officer

WU,Shifen

Telephone No. (86-10)62089928

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2015/084851

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	103576827	A	12 February 2014	None			
US	2014108639	A1	17 April 2014	CN	103729250	A	16 April 2014
CN	103338241	A	02 October 2013	None			