Information technologies in optimization process of monitoring of software and hardware status

P V Nikitin¹ A N Savinov² R I Bazhenov³ I V Ryabov²

¹ Mari State University, 1, Lenin Square, Yoshkar-Ola, 425400, Russia
² Volga State University of Technology, 3, Lenin Square, Yoshkar-Ola, 425400, Russia
³ Sholom-Aleichem Priamursky State University, Shirokaya St., 70a, Birobidzhan 679015, Russia
E-mail: petrvlni@rambler.ru

Abstract. The article describes a model of a hardware and software monitoring system for a large company that provides customers with software as a service (SaaS solution) using information technology. The main functions of the monitoring system are: provision of up-to-date data for analyzing the state of the IT infrastructure, rapid detection of the fault and its effective elimination. The main risks associated with the provision of these services are described; the comparative characteristics of the software are given; author's methods of monitoring the status of software and hardware are proposed.

1. Introduction

The software as a service (SaaS) is the users’ provision with the ability to use the vendor applications that operate on the cloud infrastructure. Access to applications is provided from various client devices through a thin client interface, for example, a web browser. The consumer does not control or manage the cloud infrastructure on which the application is running, including the network, servers, operating systems, data stores and even application settings. A possible exception is some individual application settings.

When a customer consumes any given service, rather than a software that is installed, at this very point, as a rule, he becomes a SaaS consumer. The simplest and SaaS-service used by everybody is an e-mail service - the same gmail [1].

In addition to the advantages of "cloud" approaches to the implementation of services to customers, there are a number of risks. Some of them are risks associated with the correct operation of equipment and software. If a client does not have the opportunity to get services, then the enterprises can expect negative consequences [2]. For example, a company loses customer loyalty, which can affect reputation. The company can be fined under a contract in which SLA of IT services is registered. If the company has revenue from the provision of services in the amount of time, then an idle state of such IT-service leads to direct financial losses. Therefore, in order to reduce business risks, it is recommended to implement a monitoring system in the IT infrastructure of the company [3].

The main functions of the monitoring system are: provision of up-to-date data for analyzing the state of the IT infrastructure, rapid detection of the fault and its effective elimination [4]. Also, monitoring systems can monitor the performance of IT infrastructure. It allows administrators to identify bottlenecks in the system, and also to notice a decrease in performance before users do it.
Continuous monitoring helps to maintain basic IT services in good state, helps to reduce downtime, design infrastructure upgrades and maintain a good level of service quality. Prior to automated monitoring systems, these tasks were handled by system administrators. They manually collected data on the status of systems in various non-specialized software. And sometimes information about the state was not stored anywhere and was not processed. With the departure of a specialist, all the collected data on the infrastructure, based on experience, disappeared completely.

For consumers of different levels, there are different types of monitoring systems. For small systems, as a rule, sending notifications and generalized node analysis are enough. The main functions of monitoring systems, usually, are: tracking; construction of reports; data storage; search for “bottlenecks”; visualization; automation of scripts. If the majority of the functions allocated above are implemented in the monitoring system, then it frees the administrator’s resources, as the breakdowns are eliminated faster, the diagnostics are performed accurately and multidimensionally. Also, one does not need to check the status of each node manually, and now the administrator can easily take care of scheduling changes in the infrastructure [5].

2. Review of existing automated monitoring systems
The choice of methods and objects of monitoring depends on many factors. These factors include: server and software configurations (software) installed on them, the configuration of networks operating in these networks of services, the monitoring used and the capabilities of software for monitoring. In general terms, it is possible to identify the main checks:

• check the physical availability of equipment (checking the status of RAM and the length of the disk queue);
• checking the characteristics specific to services in this particular environment (the status of services and assigned tasks, the availability of IIS sites);
• detailed check of not critical, but important characteristics of the functioning: load, productivity, etc. (availability of the virtual machine in the host network);
• checking the status of the services that are being performed (collection of OS characteristics both Windows and Linux).

Let us consider the automated systems existing on the IT markets.

1. Nagios monitoring system (originally Netsaint) - the monitoring system is mainly intended for monitoring networks and systems [6]. It is distributed for free. The system allows one to identify problems immediately after their detection, performs complex monitoring of the entire IT infrastructure, can supply interested persons with the data obtained under surveillance, has the opportunity to monitor the security of the system. These opportunities lead to a reduction in downtime and, as a result, a reduction in commercial losses. At the very beginning it was created for operating systems based on Linux, now it works equally well under Sun Solaris, FreeBSD, AIX and HPUX.

2. Zabbix automated monitoring system – a monitoring system is mainly designed for monitoring networks and systems [7], which is freely distributed. It consists of several parts. The monitoring server (the kernel) provides polls for characteristics according to a given interval, analyzes them, processes summary data, if the data exceed the preconfigured threshold values, runs scripts initiating the distribution of notifications. It has the ability to check network services remotely. It stores all configuration settings, collected historical data, statistical information in the database, usually it is MySQL or PostgreSQL. It is not installed on a server running Windows and OpenBSD. Typically, this kernel is referred to as a Zabbix server.

Comparing the capabilities of the systems described, a system Zabbix has been chosen for the automated monitoring system.

3. Designing an automated monitoring system model
Let us suppose that the data pushing center (data center) of the enterprise is in two data centers (DC), in each of them there are about 100 virtual machines, which will be covered by monitoring. Zabbix-server can work simultaneously in only one copy, so it is necessary to ensure the availability of a data
stream from one DC to another. It was decided to raise the proxy agent in each of the DCs. The database for proxy agents will be on the same machine as the agent itself. Thus, if communication between DCs is lost, agents will continue to collect indicators each in their DC, and store them in a local database (DB). Once the proxy agents have the ability to send data to the Zabbix server, they send there all the information they collected from the last communication session.

As it is stated above, the Zabbix server can work simultaneously in only one copy, so it will be launched in one of the DCs, and it is logical to store all the data in the database in the same DC, since it is considered that the closer the data handlers and the storage, the more productive the system will turn out. But leaving the data in one copy in one DC, there is a big risk of losing them entirely or access to them will not be limited at the time of the crash. Since the monitoring system should help during the crash, and not vice versa, this situation is not acceptable. One of the solutions is to configure a high-availability cluster of the database.

The level of interaction with the database is the last level of the architectural scheme of the components of the distributed remote data storage system. This level is responsible for ensuring reliable data storage, high level of their availability, and protection of the user’s information. The requirement of high availability extends to this component layer.

To ensure a high availability of a cluster from MySQL servers, replication is used as one of the techniques for scaling databases. This technique consists of the fact that data from one database server are constantly copied to another or some others. For the application, there is a possibility to use more than one server and even several servers to process all requests. Thus, it becomes possible to distribute the load from one server to several. DBMS MySQL provides several options for configuring the cluster for data replication: Master-Slave and Master-Master.

Master-Slave was chosen as the replication scheme for MySQL as more reliable, since reliability is one of the requirements for the architecture of an automated monitoring system.

Within the framework of the proposed architecture of the information system, a MySQL cluster was deployed, consisting of two machines according to the Master-Slave scheme. This scheme allows one to organize a high-availability MySQL data storage. In addition, it allows one to distribute the load on the database between several servers. The used cluster scheme consists of three machines running MySQL. One of them is a Master Server, the other two are Slave Servers. The Master Server accepts and processes incoming requests, and also maintains a binary log that synchronizes with the Slave servers. After receiving new records of the binary log, the Slave servers perform the commands specified in it and thus synchronize with the Master server.

If the Master Server fails, one of the remaining servers takes over its role. InnoDB was used as a data storage system. It differs favorably from MyISAM as it provides greater reliability when storing data. As a result, let us obtain a fault-resilient structure of the automated monitoring system Zabbix.

The automated monitoring system Zabbix has a very wide functionality, but the tools for visualization and data analysis are very complex. Each chart must be created separately to combine them into one scale. It is hard to see the data coming in real time. There is no possibility to combine graphics from different servers on one scale for comparison. This tool is very important for the analysis of both historical data and real-time data. Therefore, a convenient and lightweight tool was found for visualizing and analyzing data in the context of Grafana time. Grafana is an excellent alternative to the Zabbix dashboards. It just specializes in the visual presentation of data collected in the context of time. It allows one to work with dashboards and graphs based on data from various storage systems, including those based on the Zabbix API.

Grafana is a graphic and dashboards editor, based on data from Graphite, InfluxDB or OpenTSDB, specializing precisely in the display and analysis of information. It is not difficult, but relatively simple to install. Functionally, Grafana is a set of user dashboards, divided into user-defined heights, in which, in its turn, functional elements (graphs, html-inserts and trigger dies) can be created. Lines can be moved and renamed. Naturally, the main functional elements of the dashboards are graphs. Creating a new graph-metrics in Grafana does not require any special knowledge at all.
To integrate Grafana and Zabbix, there is a freeware module. After it is installed in the data sources, Zabbix appears for the display. An example of a dashboard in Grafana based on data from Zabbix can be seen in Figure 1.

![Image: An example of a dashboard](image)

Figure 1. An example of a dashboard

The storage requirement of a disk is obviously dependent on the number of network nodes and parameters that will be monitored. If the history of monitored parameters is planned to be kept for a long time, it should be kept in mind that at least several gigabytes are necessary for storing the history data in the database. Each Zabbix daemon process requires several connections to the database. The memory space which is required for the connection to the database depends on the database settings. Zabbix and especially the database may require significant processor resources depending on the number of monitored parameters and the selected database [8].

Main features of the developed monitoring system:
1. Collect the free disk space parameter.
   To collect the free disk space parameter, a script was written to monitor free disk space: each server is polled by WMI to generate a set of disks, after which, again using WMI, information about free and occupied space is requested. The script includes the Get-WmiCustom method, which extends the standard Get-WMIObject cmdlet with an optional parameter-the request timeout.

2. Check tasks in the task scheduler.
   To check the settings and the status of tasks of a certain type, a script was written in the Windows Task Scheduler, which forces it to launch in case of not executing the task.
   The script creates a Schedule.Service object that provides access to the Task Scheduler to manage the registered tasks. First, the Connect() method is used for the connection to the remote machine and association of all subsequent calls on this interface with this remote session. Then, the GetFolder() method is used, which gets all the contents of the registered jobs.

3. Checking the RAM.
   Checking free RAM is similar to the above-mentioned script to control free disk space: by selecting computers from Active Directory, information is collected using the WMI query. The results are processed under certain conditions. However, in addition to skipping machines from the special variable SkipHosts, the machines with the Hyper-V virtualization service installed need not be checked, because the presence of RAM on the host for virtualization is not an important indicator, only the values on the virtual machines themselves are of interest. The presence of the virtualization service is also determined via the wmi request. If the amount of available RAM on the server is less than 10%, then a message is sent to the log using the Write-Error function, and then the exit variable from the script takes the value equal to 1 so that when the execution is completed, an error is reported when this configuration step is assembled.

4. Clear logs.
   Logstash-Elasticsearch-Kibana is used to view and analyze all possible logs in the project.
4. Conclusion

Business in any sphere depends heavily on the availability and efficiency of its IT infrastructure 24/7/365. To ensure this functionality, it is necessary to identify the bottlenecks in the configuration of systems and networks in advance, as well as quickly learn about the presence of crash failure and its cause. It is customary to use monitoring systems for these needs in companies where such tracking is not possible due to specialists only.

The proposed monitoring system is developed on the basis of Zabbix. To analyze and visualize the data in the context of time, Zabbix has integrated a convenient and lightweight Grafana tool. To obtain data collection from the sources of the data center of the enterprise, private scripts have been developed.

Using this system will allow one to respond quickly to any crash failure through notifications, and preconfigured system performance visualization panels allow identifying the source of the problem and a timestamp for the beginning of degradation of characteristics. Thanks to all this, the business risk of downtime of the company’s services will significantly decrease. For example, many of the problems that eventually led to service inoperability can be determined in advance, even before users learn about it. The proposed product for monitoring servers and services, as well as automating tasks, will significantly reduce the downtime of services, improve the quality of services provided to the client by the company, and significantly improve productivity, while avoiding unjustified investments. In the future, the quality of monitoring can be improved by constant executing popular service usage scenarios.

References

[1] Lunatsi M E 2016 Power Technology and Engineering 50(4) 347-351
[2] Susarev S V 2017 Proceedings of 2017 20th IEEE International Conference on Soft Computing and Measurements 509-511
[3] Bogdanov E A 2016 Biomedical and Pharmacology Journal 9(2) 483-489.
[4] Indhumathi V 2016 International Journal of Applied Engineering Research 10(24) 43993-44000
[5] Cao Z-W 2017 Wuli Xuebao 66(2) 020301
[6] Nagios 2018 [Electronic resource]: Nagios Documentation // URL: http://www.nagios.org/ (the date of reference 12.02.2018).
[7] Zabbix Documentation 2018 [Electronic resource]: Zabbix documentation for version 2.0. // URL: http://www.zabbix.com/documentation/2.0 (the date of reference 12.02.2018).
[8] Dalle Vakke A. Zabbix. 2017 Code of practice (М.: DMK Press)