15. Warren, P. Knowledge Management and the Semantic Web: From Scenario to Technology [Text] / P. Warren // IEEE Intelligent Systems. – 2006. – Vol. 21, Issue 1. – P. 53–59. doi: 10.1109/mis.2006.12
16. Uschold, M. Ontologies: principles, methods and applications [Text] / M. Uschold, M. Gruninger // The Knowledge Engineering Review. – 1996. – Vol. 11, Issue 02. – P. 93. doi: 10.1017/s0269888900007797
17. Wagner, C. Wiki: A technology for conversational knowledge management and group collaboration [Text] / C. Wagner // Communications of the Association for Information Systems. – 2004. – Vol. 13. – P. 264–289. – Available at: http://aisel.aisnet.org/cgi/viewcontent.cgi?article=3238&context=cais
18. Spenser, S. Kompetencyy na rabote [Text] / S. Spenser, L. Spenser. – Moscow: HIPPO, 2005. – 384 p.
19. Rogushina, J. Ontology-based competency analyses in new research domains [Text] / J. Rogushina, A. Gladun // Journal of Computing and Information Technology. – 2012. – Vol. 20, Issue 4. – P. 277. doi: 10.2498/cit.1002034
20. Rogushina, J. Use of the Ontological Model for Personification of the Semantic Search [Text] / J. Rogushina // International Journal of Mathematical Sciences and Computing. – 2016. – Vol. 2, Issue 1. – P. 1–15. doi: 10.5815/ijmsc.2016.01.01

Розроблено структуру планувальника розподіленої комп'ютерної системи, яка підтримує захищену обробку даних. Запропоновано структуру централізованого планувальника, який, взаємодіючи з локальними агентами даних обчислювальних вузлів, визначає параметри вузлів системи та підбирає ресурси для задачі з встановленими вимогами щодо захищеності. Подано результати досліджень застосування класичного та адаптивного механізмів управління захищеністю в РКС на прикладі Grid-системи

Ключові слова: розподілена комп'ютерна система, управління обчислювальними ресурсами, планування задач, моніторинг параметрів обчислювального вузла

Разработана структура планировщика распределенной компьютерной системы, поддерживающей защищенную обработку данных. Предложена структура централизованного планировщика, который, взаимодействуя с локальными агентами данных вычислительных узлов, определяет параметры узлов системы и подбирает ресурсы для задачи с заданными требованиями по защищенности. Проведены исследования по использованию классического и адаптивного механизмов управления защищенностью в распределенной компьютерной системе на примере Grid-системы

Ключевые слова: распределенная компьютерная система, управление вычислительными ресурсами, планирование задач, мониторинг параметров вычислительного узла

UDC 004.75
DOI: 10.15587/1729-4061.2017.91271

THE SCHEDULER FOR THE GRID-SYSTEM BASED ON THE PARAMETERS MONITORING OF THE COMPUTER COMPONENTS

Hu Zhenbing
PhD
School of Educational Information Technology
Central China Normal University
Louyu str., 152, Wuhan, China, 430079
E-mail: hzb@mail.ccnu.edu.cn
V. Mukhin
Doctor of Technical Sciences*
E-mail: v.mukhin@kpi.ua
Ya. Kornaga
PhD**
E-mail: y.kornaga@kpi.ua
O. Herasymenko
Assistant
Department of Networking and Internet technologies
Taras Shevchenko National University of Kyiv
Volodymyrska str., 60, Kyiv, Ukraine, 01033
E-mail: oksgerasymenko@gmail.com
Yu. Bazaka
Postgraduate student**
E-mail: yura.bazaka@gmail.com
*Department of Computing Technics***
Department of Technical Cybernetics*
***National Technical University of Ukraine
«Igor Sikorsky Kiev Polytechnic Institute»
Peremohy ave., 37, Kyiv, Ukraine, 03056

1. Introduction

Since the emergence up to the present, distributed computer systems have undergone substantial changes. At the same time, resource allocation and management in such systems remain an important problem and researchers are working towards its solution. There are various DCS resource management approaches, most of which being focused on time parameters of task processing by the system and maintenance costs. However, in some cases during task

placement for execution, the distributed system resource security is important for the user. Some DCS performance loss as a result of security provision is inevitable. So, the resource management problem becomes more complicated in this case. This is due to the fact that provision of an optimum combination of the task execution time parameters and the security parameters of the involved system components is required in the compute nodes (CN) selection.

Let us consider this task in the context of the Grid system, which is a type of distributed computer systems. The Grid system is known to have a service quality indicator, which is a certain set of the distributed system operation parameters. The factors influencing the task execution time parameters values include CN performance and data transmission rate in the distributed system communication channels.

Let us create the DCS resource management task as follows: to create a set of resources from the available distributed system resources for task execution, which would have the optimum ratio of the "task execution time" – "resource security level" parameters. The resource security level is specified by users according to their own requirements. It must be taken into account that in case the user requires high-security resources, but they are unavailable in the system, the task has to be rejected and the user should be notified.

Existing mechanisms for the DCS parameters analysis and resource management do not allow comprehensive accounting for the factors such as DCS performance and security level of information processed. Implementation of this approach will allow singling out the DCS resource regions with sufficient security and performance levels. The main problem in creating such tools is that the current scheduling mechanisms have to be adapted to the factors integration requirements. This actually requires development of the modified scheduler structure and integration of the functional units, focused on the adaptive resource management mechanism, which is one of the key issues in the paper.

2. Literature review and problem statement

Currently, there are some Grid task schedulers modifications, which are significantly different in architectural decisions and functionality. In [1, 2] the schedulers construction principles and implementation approaches for Grid have been considered in detail, the operation algorithms have been offered. However, despite a large number of works devoted to the Grid system resource management mechanisms, these mechanisms still need to be improved [2].

One of the approaches used in the distributed system resource management is based on the necessity of accounting for communication environment parameters, as well as transmitted data volume and location in the scheduling. The emergence and development of this approach are due to the fact that distributed systems are increasingly being used for data storage. The resource management mechanisms, based on this approach are becoming more common. So, in [3] task scheduling in Cloud considering the channel capacity has been presented. In [4] heuristic task scheduling algorithms in Grid considering network communication have been provided. The authors of [5] have presented the scheduling algorithm that accounts for the task input data allocation in the distributed system. In [6] the approach that considers the network and resource status dynamics in task scheduling has been proposed. The authors of [7] present task scheduling with accounting for the network channel capacity for intensive data exchange applications. In [8] the optimization method for job scheduling and load balancing in Grid systems based on genetic algorithms has been proposed.

In [3–8] the algorithms based on this approach have been presented.

One of the most common DCS resource management approaches is the approach based on providing the required quality of service. The QoS indicator is expressed in quality assessment of parameters such as total task execution time, delay, task execution price, packet loss rate, system performance and reliability [9]. However, this list is not exhaustive. The scheduling mechanisms aimed at providing the required QoS indicator have been presented in a wide range of works. The work [9] offers task scheduling in Cloud, computing the execution time and thereby setting the task priority. In [10] heuristics have been used to provide the required QoS. An adaptive QoS (AQoS) scheduling algorithm in service-oriented Grid environments has been presented in [11]. The authors of [12] offer to perform so-called meta-scheduling preliminary to achieve the desired QoS.

The DCS resource management approach aimed at ensuring secure task execution and thus secure data processing should also be highlighted. So, in [13] the task scheduling mechanism, considering the system resource security level for the distributed computing environment with intensive data exchange has been presented. The disadvantage of this approach is that the resource security level variation in the distributed environment functioning process is not considered. For the Cloud environment, the workflow-based trust-driven task scheduling strategy has been presented [14]. The strategy considers the task execution time, service cost, service reliability and security, and the workflow-based application is a set of atomic interdependent tasks, whose execution data are stored in the Cloud environment services.

Literature review has shown that secure data processing in DCS is implemented using the classical resource security management mechanism. Introduction of the distributed system adaptive resource security management mechanism will inevitably affect the DCS functioning. Therefore, the paper aims to investigate the adaptive resource security management mechanism impact on the distributed system performance.

3. Aims and objectives of the study

The aim of the study is to develop the scheduler, which supports the provision of appropriate QoS with the required data security.

To achieve this goal, the following objectives have been formulated:

– to develop the structure of the centralized scheduler, which supports secure data processing;

– to investigate the classical security management mechanism impact on system performance;

– to investigate the adaptive security management mechanism impact on system performance;

– to investigate the compute nodes parameters monitoring system impact on system performance.

4. Materials and methods

The research employs the methods of the scheduling theory and load balancing in distributed computer systems, computer systems security monitoring systems theory, complex systems design theory, automated control theory, observation results processing methods.

The research of distributed systems in actual practice is not always possible in view of being expensive and time-consuming. So, simulation as a way to solve this problem is used. There are several distributed systems simulation environments, such as GridSim (Australia) [15], ALEA2 (the Czech Republic) [16]. GridSim is a basic set of the components, developed in the concept of object-oriented programming, the use of which allows building models of distributed systems with different configurations. The ALEA2 environment is also based on GridSim, but it is designed to study the cluster behavior and loading. The GridSim environment was chosen for the experiment.

For the research, a special software package that uses GridSim library (Australia) was developed [15]. The GridSim component library does not allow simulation of secure data processing in DCS. So, the main objective of this package was secure data processing in the Grid system model. The developed mechanism supports both the classical and the adaptive distributed system resource security management mechanism. In addition, simulation of the status monitoring system of compute nodes as one of the main components of the adaptive DCS resource security management mechanism was implemented. To develop the Grid system model, the existing GridSim library classes were modified, and some new classes for data collection and input data preparation were developed.

Let us briefly present modification ways of GridSim library components for the model building. In [15] the existing GridSim library class diagram has been presented.

The input data for simulation is the computational complexity of a task in MI (million instructions) and time of its occurrence in the system. The data are read from the input data file, where characteristics of each task are recorded in a new line and separated by a semicolon. Also, the software package employs the possibility to create a file with the task parameters. For this, the GridSimRandom class from the GridSim package is used.

The main goal of the task implementing class modification is to add the characteristic of security level, which is the requirement of a task to compute nodes. To do this, the new TaskWidthSecureReq class, which inherits the Gridlet class of the GridSim package was developed. In the new class, the appropriate class variable and methods to access it were added.

When reading the task characteristics from the file and creating a set of tasks, task requirements to the resource security level can be generated using the GridSimRandom class ranging from 0.1 to 0.9, or specified by the user.

Compute node. The computing resource (GridResource class) in the GridSim package is formed of a set of machines or computers (one or more), each consisting of a set of processors (one or more). We call a computing resource a compute node no matter how many computers and processor elements it contains. The ResourceCharacteristics class is used to preserve the resource characteristics, as well as by the scheduler when storing data on resources of the entire system. Because each compute node (CN) is characterized by an additional trust level parameter, the new GridResourceWithSecure class, which inherits the GridResource class and the ResourceCharacteristicsWithSecure class, which inherits the ResourceCharacteristics class were created.

As noted above, introduction of the adaptive DCS security management mechanism provides for installing additional software on CN. For the CN computing load additional constant simulation, the new MachineWithMonitoring class, which inherits the Machine class was implemented. The new class has an additional class variable, which represents the additional computing load on CN in MI. In the research of the monitoring system effect on the entire DCS operation, in all the MachineWithMonitoring class examples this variable had the same value regardless of computing power because all computers had the same additional monitoring system software.

One of the main Grid system model components, which is directly responsible for resource selection and task placement on resources is implemented by the TheScheduler class. When a new task is received, selection of free CN with appropriate trust level and performance is carried out. If no CN is suitable, the task is added to the task queue.

Upon the task processing completion, the CN generates an availability signal to the scheduler. Then, the scheduler sends a task for processing, which at the moment is the first in the queue with the relevant node trust level requirements.

Simulation results gathering and output data format.

The Grid system operation data gathering is implemented using several parameters, namely:

– task queue length in certain timepoints;
– task occurrence time in the system;
– task processing start time;
– task processing completion time;
– the number of available and backlogged CN in certain timepoints and the corresponding total computing power in MIPS (million instructions per second), etc.

The GridSim package has the Stat class, designed to output statistical data to a file. However, analysis of data from the file, in which they are recorded, with specialized programs is inconvenient because of the specific output format. Therefore, the software package implements the additional PeriodicGathering class, designed to gather the necessary data and the StatSaving class to save the gathered data in various files according to their purpose. Output data are separated by a semicolon and do not contain unnecessary comments, which allows using specialized software for further processing.

Detailed diagnostic messages are displayed during software system operation, which allows the simulation correctness monitoring. A more detailed description of the experiments is presented below (paragraph 8), because this stage has not yet submitted all necessary information about the development.

5. Determination of requirements for the scheduler and operation assessment parameters

Requirements for the scheduler depend on many factors, including the system goal, types of tasks, resource performance, quality of service (QoS) parameters, etc. Determination of requirements for the system scheduler and operation assessment parameters is an important design stage.

The scheduler has to implement a certain set of features that are considered actually typical for Grid systems at

present, including tasks migration and rescheduling, control points, task execution time interval pre-order, etc.

The developed scheduler allows for system operation parameters such as performance and processed data security level, which are mutually conflicting in general. As a result, the developed scheduling mechanism will allow creating a high-performance distributed computer system, which supports the required security level of the data processed.

Based on the obtained general concept of the scheduler, its functionality can be expanded in the future. Moreover, this approach can be appropriately modified for use in other DCS, such as clusters or Cloud systems.

Since there are several types of DCS, each having its own construction, management, and operation peculiarities, it is necessary to clarify what we mean by DCS in the paper. In the research, we represent DCS as a set of heterogeneous, geographically distributed CN, which use computer network as a data transmission system and join together (interact) to perform complex computing tasks. A complex computing task means the task with a high degree of parallelism. The task is a workflow, but the task components placement on the system compute nodes is not the research subject. The QoS indicator may consider a set of parameters. QoS in this paper means the time spent on task execution and ensuring appropriate data processing security.

Let us present the following requirements for the developed scheduler:

– the scheduler is developed based on the centralized Grid system management approach. Let's call it further the meta-scheduler (or global scheduler);

– the scheduler is focused on scalable tasks. Execution of other tasks is possible, but the issues of the task placement as a workflow on the system compute nodes is beyond the scope of the research;

– QoS requirements refer only to the resource security level and the task execution time, the service cost calculation is not performed.

6. Task execution start in the presented DCS

Sending a task for execution requires the user to set parameters of the DCS resources on which it will be performed. As noted above, the meta-scheduler has to provide the required QoS, which in this case is the task execution time and the resource security level. System resource security is specified by the user in the range from 0 to 1 because it is the user who determines the data importance. The task execution speed depends on many distributed system parameters, including the computing resource performance, data transmission rate in communication channels, system components reliability and so on. The user can set the task execution time, or there is another option – the system will use the DCS resources with the best performance for speedy task execution. DCS services valuation in accordance with the data processing security level and the required task execution time is beyond the scope of the research.

7. Meta-scheduler structure and operation

The meta-scheduler is designed to service a task queue and select resources for task execution in accordance with the user's requirements, as well as to monitor the task execution process. The resource selection is based on determination of functional DCS parameters such as performance, security, and data transmission rate. Functional characteristics of the system are determined on the basis of the data received from each CN from the local data agent (LDA), and also the task parameters such as computing complexity in MIs and the amount of input data. In addition, the user specifies the security level from 0 to 1 and the maximum time of task execution. LDA refers to the specialized software, which monitors the compute node status by parameters such as performance, data transmission rate, and security level, and also sends them to the meta-scheduler during interaction.

7. 1. Meta-scheduler operation

When a new task is received, the current DCS status is assessed by sending requests for information from each device (if the current system status assessment occurred relatively recently, this step can be omitted). Based on the obtained data, task parameters, assessed by the meta-scheduler automatically, and the user's requirements for the system resource security, DCS functionality parameters computations are carried out. As a result, the number of system nodes required for task execution with the specified QoS parameters is determined.

Further, available CN are selected to create a set of resources for task execution; in the simplest case, this can be done using the resource security mask. In case all resources are available, a task is placed for execution. Otherwise, the task priority is determined and it is placed in a queue according to its priority. If resources became available after the task execution, the next task is selected from the queue. Task rescheduling is also possible if the priorities of all tasks, which are executed at the moment are lower than the priority of the task that is in the beginning of the queue. Moreover, rescheduling can occur in the event of the system CN failure.

It is important to note that the system functionality reassessment is necessary in each rescheduling cycle and for each task in the queue in view of the constantly changing distributed system parameters.

7. 2. A meta-scheduler block diagram

The meta-scheduler block diagram was developed in accordance with the functional characteristics (Fig. 1).

Queue control unit receives input tasks and, in accordance with the scheduler settings, computes their priority and adds to the queue. Its functions also include task rescheduling. In case of rescheduling, the tasks, execution of which was suspended are also added to the queue. The priority of such tasks raises.

Task queue is designed to place tasks with their parameters according to the priority, computed by the queue control unit. It is important to note that the queue not only includes a task itself, but also all the information related to the user's requirements for DCS resources for its execution.

Scheduler settings contain the queue formation and task priority computing parameters.

DCS functionality parameters computing unit is responsible for the compute nodes and LDA interaction and the system functionality parameters determination. The obtained system functionality parameters are used to determine the number of CN needed for task execution.

Resource selection unit is responsible for identification of specific DCS nodes, which will be used to place a task for execution. The number of nodes is determined by the *DCS functionality parameters computing unit*. Nodes are selected

by masking, the mask parameters are set as follows: security is indicated as a level of trust to the node in the range from 0 to 1, and the most high-performance CN are selected.

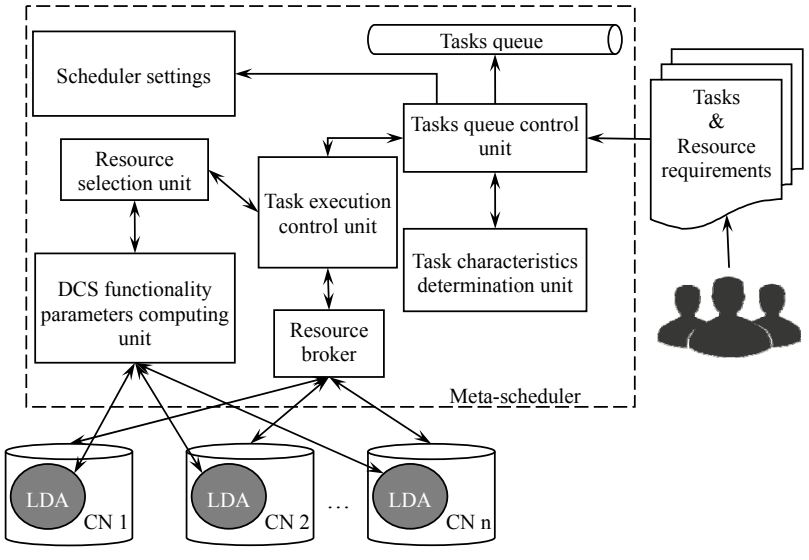

Fig. 1. The DCS meta-scheduler block diagram

Task execution control unit selects tasks from the queue beginning and, by interacting with the resource selection unit, receives a list of the resources, selected for task execution. It places a task for execution through the resource broker. If a task, which has priority over all tasks executed at the moment, is received when viewing the queue, it suspends execution of tasks, keeping their status and data, and places new tasks on resources after rescheduling.

Task characteristics determination unit is designed to determine the number of task computations in MIs.

Resource broker is responsible for direct interaction between the scheduler and resources, that is, identifies available resources, provides resources in accordance with the task execution request, releases resources after task execution.

8. DCS parameters experimental results

8. 1. Experimental set up

For the analysis of the developed approach to the distributed system resource management, experimental studies were made. It is important to note that experiments were aimed at studying the results of implementation of the adaptive system compute node trust level management mechanism. Development and research of the DCS resource management optimization mechanism based on the proposed approach go beyond the scope of this paper.

Thus, let us identify the basic research directions:

– assessment of the adaptive security management benefits by taking into account the CN trust level in task execution;

– analysis of the CN status monitoring system impact on the DCS performance.

8. 2. Research of the compute node monitoring system impact on DCS performance

It is proposed to place additional software on CN for the compute nodes status monitoring. Extra software reduces their performance and the performance of the entire distributed system. Thus, it is important to investigate the additional load possible effect on the entire system performance.

In order to determine the system performance loss, it is necessary to perform DCS simulation at a certain set of tasks considering the monitoring system operation and without it at other equal conditions. By varying the compute nodes performance share, allocated for the monitoring system operation, let's consider the overall DCS performance variation. It is important that the task scheduling algorithm that can repeatedly place tasks on system resources according to the same scheme was applied in the simulation. Simple FCFS algorithm (First Come First Served) was used in the research.

The aim of the experiment is to determine the DCS operation parameters in case of extra software (monitoring system) on the system compute nodes. To do this, let's first simulate task execution without additional computing load on the DCS CN, and then we introduce load for each node. We used the average task waiting time in the queue, the average task residence time in the system, and the average task queue length as the system performance evaluation parameters. Importantly, we decided to take a flow of input tasks with an intensive task reception for the experiments in order to more fully estimate the time delays occurring in the system.

The DCS model making assumes the availability of the system components characteristics. The simulation was performed on the system of 5 uniprocessor compute nodes, the parameters are taken from the performance tests presented in [15]. Three sets of tasks: 150, 500 and 1000 tasks in a set were used in the experiments. Since the aim of the experiment is to study the system functioning with intensive task flow, the task set formation was made by software generation of the task length in MIs, volumes of input and output data, as well as time intervals between tasks. Task flow characteristics are shown in Table 1. Task flows in real systems can be found in [17, 18].

Table 1

Input task flow characteristics for the experiment

Characteristic	Flow 1	Flow 2	Flow 3
Number of tasks	150	500	1000
Receipt intensity, u/s	0.02	0.02	0.02
Minimum task length, MI	2071	2750	3918
Maximum task length, MI	3476761	3490799	3495453
Average task length, MI	1771381	1677297	1788710

The research for each set of tasks was performed in three versions: no monitoring system (experiment 1), monitoring system utilizes 5 % of the node computing power (experiment 2), and monitoring system utilizes 25 % of the node compute power (experiment 3). The task receipt order and time in a set of three experiments were the same; the scheduling and placement algorithm was the simplest – the first free resource – also did not change during the research. Average values of the experiments are shown in Table 2. Fig. 2 shows bar graphs with a graphical representation of experimental results.

Table 2

Experimental results

Task flow for experiment	Type of experiment	Average waiting time, h	Average residence time in the system, h
Flow 1	Experiment 1	15.86	17.08
	Experiment 2	17.19	18.50
	Experiment 3	25.71	27.64
Flow 2	Experiment 1	52.55	53.70
	Experiment 2	57.05	58.28
	Experiment 3	85.26	87.07
Flow 3	Experiment 1	115.36	116.58
	Experiment 2	125.08	126.01
	Experiment 3	186.81	188.76

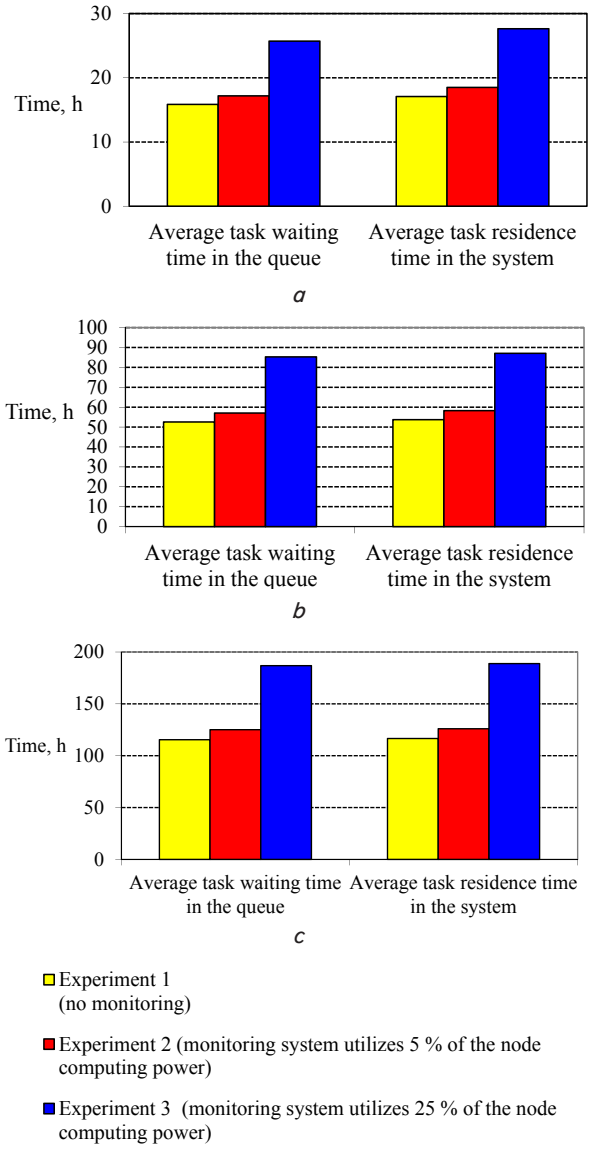

Fig. 2. Experimental results on the CN status monitoring system effect on the DCS performance: *a* — flow 1 (150 tasks); *b* — flow 2 (500 tasks); *c* — flow 3 (1000 tasks)

Experimental results indicate that additional software placement on the distributed system CN for their status monitoring puts an additional load on the entire DCS computing power. The experiments demonstrate that the computing power required by the CN processor to ensure the monitoring system operation is also important. Thus, according to the experiments, the average task waiting time in the queue and the average task residence time in the system increased by 8 % at 5 % processor computing power. In case of 25 % CN processor computing power for monitoring, the average task waiting time in the queue and the average task residence time in the system is increased greatly, which indicates a significant loss of the entire DKS performance. If we consider the allowable expenditure for security to be at the level of 10 %, then we can talk about the CN status monitoring system admissibility to ensure secure data processing in the distributed system. On the other hand, the monitoring software selection and development have to be based on functional features and purpose of the distributed system. It should also be noted that the average task waiting time in the queue and the average task residence time in the system greatly depend on the DCS CN set and the input task flow parameters. In these experiments, they are taken in order to demonstrate the CN status monitoring system effect on the DCS performance and are not crucial for the system functionality assessment.

8. 3. Research of the adaptive DCS compute node trust level management mechanism

The adaptive trust level management mechanism is based on monitoring the actions of CN (the user who owns the CN data) in DCS and allows the system to be more responsive to security threats. The main thing in the actions monitoring process is to identify cases of the user's unauthorized access or attacks. The review of methods and means of identifying such facts is presented in [19]. Network traffic anomaly detection is basically applied for this purpose [20]. Traffic analysis can be done by various methods, including data mining or artificial intelligence methods [21]. More details on intrusion detection systems can be found in [22].

At the same time, it should be noted that the user actions are not always unlawful, and may be caused by an accident or carelessness. The adaptive mechanism accounts for these factors. More details on the adaptive trust level management mechanism are available in [23].

Since this mechanism allows identifying incorrectly behaving DCS CN and then excluding them from the system, the bulk (except for newly connected) of the distributed system nodes will have a high level of trust after some time. In that case, when the system receives tasks with high data security requirements, DCS can provide a greater number of CN that are suitable for executing these tasks. Accordingly, the task waiting time in the queue and the task residence time in the system are reduced. This experiment is designed to demonstrate the foregoing.

Let's consider the task flow execution in the distributed system with the classical security mechanism and in the distributed system with the adaptive trust level management mechanism. In the first case, each distributed system compute node is characterized by attribute such as trust level that remains unchanged during operation. The task, entering the system cannot be executed on the CN, the trust level of which is lower than that specified in the task owner's requirements. In the second case, the level of trust to a node is changed due to attacks and incorrect actions detected during monitoring, emanating from the

node, and the trust level can either decrease or increase. The node selection requirements are similar to the first case.

Using the GridSim environment, the DCS model was developed, and the task flow execution with the classical and adaptive trust level management mechanisms was investigated. The DCS compute node parameters are similar to those described in the previous experiment. The CN trust level values with the classical mechanism amounted to 0.3; 0.4; 0.5; 0.6 and 0.7. When applying the adaptive management mechanism, we believe that the nodes were just connected to the DCS, so the initial trust level of all nodes is low and amounts to 0.1. Tasks were generated by the random number generator. The task flow characteristics are the following: the number of tasks – 150, the receipt intensity– $0.5 \cdot 10^{-3}$ u/s, the minimum task length – $3.24 \cdot 10^{11}$ MI, the maximum task length –$1.62 \cdot 10^{12}$ MI, the average task length – $9.59 \cdot 10^{11}$ MI. For convenient presentation of further results, all tasks have the same requirements for secure data processing, namely, the CN should have the trust level of at least 0.5.

In the adaptive CN trust level management mechanism simulation, the number of attacks on the node was also generated using the random number generator. The compute node security system is able to repel 6 security incidents. If more, the CN trust level decreases. If the number of attacks is less than 6, the trust level increases, but it cannot take a value above 0.9 (value 1 is not used since there are no absolutely secure resources).

Table 3 shows the average task waiting time in the queue and the average task residence time in the system with different DCS CN trust level management mechanisms. Fig. 3 presents a bar graph based on the data from Table 3. As can be seen from the experimental results, the use of the adaptive security level management mechanism makes the average task waiting time in the queue and the average task residence time in the system significantly lower than in case of the classical security level management mechanism.

Table 3

The average task service time in DCS with different trust management mechanisms

Indicator \ Management mechanism	Classical	Adaptive
Average waiting time, h	6.07	2.15
Average residence time in the system, h	7.79	3.75

Fig. 4 shows the dynamics of the task queue length in the DCS with different CN trust management mechanisms. As in the previous experiment, we used the simplest service algorithm FCFS (First Come First Served) in this research.

Fig. 4 shows that the queue in the DCS with the adaptive trust level management mechanism initially is longer than in that with the classical trust level management mechanism. This is due to the fact that the CN initially received a low level of trust, which was lower than that required for task execution. During the system operation process, the CN received such a level of trust that allowed them to execute the tasks entering the system, so the queue length gradually decreases. The queue for the system with the classical trust level management mechanism grows steadily as the system resources are not sufficient for task processing (two resources have the trust level lower than that required for task execution).

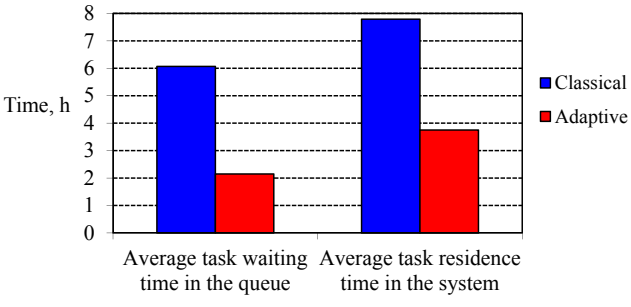

Fig. 3. Comparison of the average task waiting time in the queue and the average task residence time in the system with different CN trust management mechanisms

Consistently high trust level of computing resources in the DCS with the adaptive trust level management mechanism is due to the fact that the detected problematic nodes are subsequently removed from the system. At the same time, the remaining nodes are reliable, so with time they acquire a high trust level, which may slightly decrease for a while and then return to a high level. This explains a small task queue length after the DCS transient mode ended (in the beginning).

Given all the above, it is regular that the DCS with the adaptive trust level management mechanism has executed the task flow much earlier than that with the classical trust level management mechanism (Fig. 4).

Similar conclusions can be made from Fig. 5, which shows the DCS compute nodes utilization rate with different CN trust management mechanisms.

As can be seen from Fig. 5, the load on the DCS with the classical trust level management mechanism is only 60 % (3 of 5 nodes), while the DCS with the adaptive trust level management mechanism mainly operates at the resource utilization rate above 60 %.

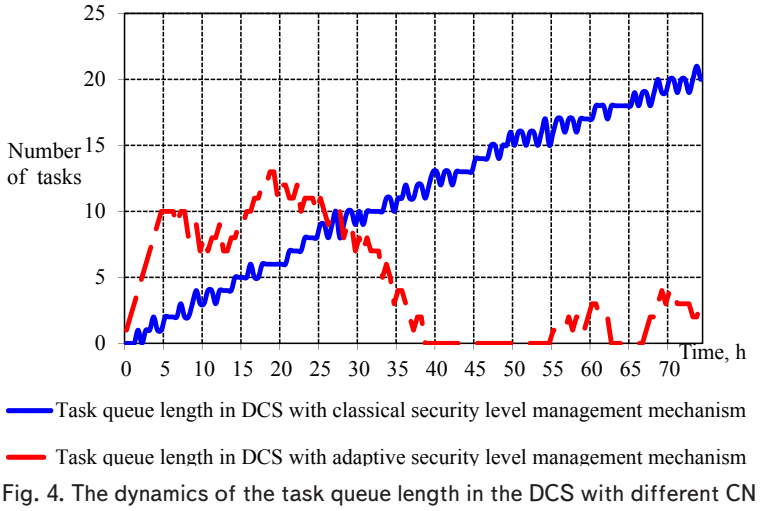

Fig. 4. The dynamics of the task queue length in the DCS with different CN trust management mechanisms

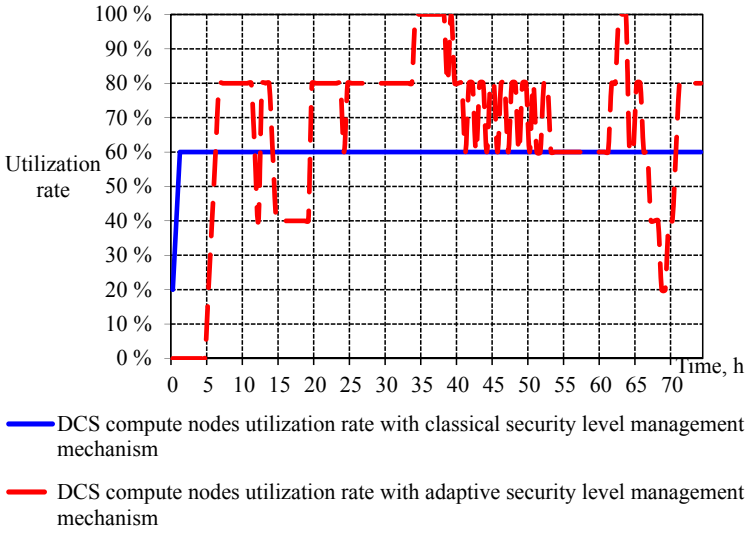

Fig. 5. The DCS compute nodes utilization rate with different CN trust management mechanisms

8. 4. Discussion of the research results on the adaptive security management mechanism effect on the DCS parameters

The research revealed that the DCS with the proposed adaptive security management mechanism demonstrates higher system performance compared to that with the classical security management mechanism. At the same time, the task queue length increases at the DCS initial functioning stage with the adaptive security management mechanism due to the distributed system transient operation mode.

Also, the analysis of the monitoring system impact on the DCS performance was made in the research. The research has shown that the monitoring system to some extent reduces the DCS performance, while supporting a sufficiently high security level of the information processed.

The research is universal and can be used for analysis of a wide range of the DCS parameters in the secure data processing mechanism implementation.

A further area of research is the DCS resource allocation intellectualization, considering the QoS requirements, as well as the decentralized scheduler development for the DCS with secure data processing support.

9. Conclusions

1. The DCS task scheduler block diagram based on the centralized approach, which supports secure data processing on the basis of the adaptive distributed system resource security management mechanism was developed. Implementation of the adaptive security management mechanism requires the specialized software on the compute nodes through which the scheduler receives the compute node parameters and computes the DCS parameters. On the basis of the parameters computation results, resources for task execution are selected, taking into account the user's requirements.

2. The comparative analysis of the classical and adaptive security management mechanisms impacts on system performance was made. The research has shown that the average task waiting time in the queue and the average task residence time in the system with the adaptive security level management mechanism is 2.8 and 2.1 times lower, respectively. According to the experiments, the average task queue length in the distributed system with the adaptive security level management mechanism is 2.4 times less than the average task queue length in DCS with the classical security level management mechanism.

3. The analysis of the CN parameters monitoring system on the DCS performance was made. It is shown that the monitoring system can significantly reduce the DCS performance. Thus, according to the experiments, in case of 25 % load on the DCS CN from the monitoring system, the average task waiting time in the queue and the average task residence time in the system increase by 62 % compared with a situation where monitoring is not performed. This is because the fourth of the system computing power is constantly used by the monitoring system, thus increasing the time required for task execution.

References

1. Zhu, Y. A Survey on Grid Scheduling Systems [Text] / Y. Zhu, L. M. Ni. – Technical Report # SJTU_CS_TR_200309001. – Shanghai Jiao Tong University, 2013. – 41 p. – Available at: http://www.cs.sjtu.edu.cn/~yzhu/reports/SJTU_CS_TR_200309001.pdf
2. Qureshi, M. B. Survey on Grid Resource Allocation Mechanisms [Text] / M. B. Qureshi, M. M. Dehnavi, N. Min-Allah, M. S. Qureshi, H. Hussain, I. Rentifis et. al. // Journal of Grid Computing. – 2014. – Vol. 12, Issue 2. – P. 399–441. doi: 10.1007/s10723-014-9292-9
3. Lin, W. Bandwidth-aware divisible task scheduling for cloud computing [Text] / W. Lin, C. Liang, J. Z. Wang, R. Buyya // Software: Practice and Experience. – 2012. – Vol. 44, Issue 2. – P. 163–174. doi: 10.1002/spe.2163
4. Caminero, A. Network-aware heuristics for inter-domain meta-scheduling in Grids [Text] / A. Caminero, O. Rana, B. Caminero, C. Carrion // Journal of Computer and System Sciences. – 2011. – Vol. 77, Issue 2. – P. 262–281. doi: 10.1016/j.jcss.2010.01.006
5. Jin, J. BAR: An Efficient Data Locality Driven Task Scheduling Algorithm for Cloud Computing [Text] / J. Jin, J. Luo, A. Song, F. Dong, R. Xiong // 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. – 2011. doi: 10.1109/ccgrid.2011.55
6. Yang, C.-T. Network Bandwidth-aware job scheduling with dynamic information model for Grid resource brokers [Text] / C.-T. Yang, F.-Y. Leu, S.-Y. Chen // The Journal of Supercomputing. – 2008. – Vol. 52, Issue 3. – P. 199–223. doi: 10.1007/s11227-008-0256-3
7. McClatchey, R. Scheduling in Data Intensive and Network Aware (DIANA) Grid Environments Architecture [Electronic resource] / R. McClatchey, A. Anjum, H. Stockinger, A. Ali, I. Willers, M. Thomas // Available at: https://arxiv.org/ftp/arxiv/papers/0707/0707.0862.pdf

8. Singh, R. Cuckoo Genetic Optimization Algorithm for Efficient Job Scheduling with Load Balance in Grid Computing [Text] / R. Singh // International Journal of Computer Network and Information Security. – 2016. – Vol. 8, Issue 8. – P. 59–66. doi: 10.5815/ijcnis.2016.08.07
9. Wu, X. A Task Scheduling Algorithm based on QoS-Driven in Cloud Computing [Text] / X. Wu, M. Deng, R. Zhang, B. Zeng, S. Zhou // Procedia Computer Science. – 2013. – Vol. 17. – P. 1162–1169. doi: 10.1016/j.procs.2013.05.148
10. Chauhan, S. S. A Heuristic for QoS Based Independent Task Scheduling in Grid Environment [Text] / S. S. Chauhan, R. C. Joshi // 2010 5th International Conference on Industrial and Information Systems. – 2010. doi: 10.1109/iciinfs.2010.5578725
11. Ang, T. F. Adaptive QoS scheduling in a service-oriented grid environment [Text] / T. F. Ang, T. Ch. Ling, K. K. Phang // Turk Journal of Electronic Engineering & Computer Science. – 2012. – Vol. 20, Issue 3. – P. 413–424.
12. Conejero, J. QoS Provisioning by Meta-scheduling via advance within SLA-based Grid Environments [Text] / J. Conejero, L. Tomas, B. Caminero, C. Carrion // Computing and Informatics. – 2012. – Vol. 31. – P. 73–88.
13. Liu, H. Swarm scheduling approaches for work- ow applications with security constraints in distributed data-intensive computing environments [Text] / H. Liu, A. Abraham, V. Snasel, S. McLoone // Information Sciences. – 2012. – Vol. 192. – P. 228–243. doi: 10.1016/j.ins.2011.12.032
14. Yang, Y. L. Trust-Based Scheduling Strategy for Cloud Workflow Applications [Text] / Y. L. Yang, X. G. Peng, J. F. Cao // Informatica. – 2015. – Vol. 26, Issue 1. – P. 159–180. doi: 10.15388/informatica.2015.43
15. Buyya, R. GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing [Text] / R. Buyya, M. Murshed // Concurrency and Computation: Practice and Experience. – 2002. – Vol. 17, Issue 13-15. – P. 1175–1220. doi: 10.1002/cpe.710
16. Klusacek, D. Alea 2: job scheduling simulator [Text] / D. Klusacek, H. Rudova // Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques. – 2010. doi: 10.4108/icst.simutools2010.8722
17. Logs of Real Parallel Workloads from Production Systems [Electronic resource]. – Available at: http://www.cs.huji.ac.il/labs/parallel/workload/logs.html
18. The Grid Workloads Archive: The Grid Workloads Datasets [Electronic resource]. – Available at: http://gwa.ewi.tudelft.nl/datasets/
19. Bhuyan, M. H. Network Anomaly Detection: Methods, Systems and Tools [Text] / M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita // IEEE Communications Surveys & Tutorials. – 2014. – Vol. 16, Issue 1. – P. 303–336. doi: 10.1109/surv.2013.052213.00046
20. Heidarian, Z. Intrusion Detection Based on Normal Traffic Specifications [Text] / Z. Heidarian, N. Movahedinia, N. Moghim, P. Mahdinia // International Journal of Computer Network and Information Security. – 2015. – Vol. 7, Issue 9. – P. 32–38. doi: 10.5815/ijcnis.2015.09.04
21. Khobzaoui, A. Intrusion Detection with Multi-Connected Representation [Text] / A. Khobzaoui, A. Yousfate // International Journal of Computer Network and Information Security. – 2016. – Vol. 8, Issue 1. – P. 35–42. doi: 10.5815/ijcnis.2016.01.05
22. Liao, H.-J. Intrusion detection system: A comprehensive review [Text] / H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, K.-Y. Tung // Journal of Network and Computer Applications. – 2013. – Vol. 36, Issue 1. – P. 16–24. doi: 10.1016/j.jnca.2012.09.004
23. Mukhin, V. Ye. The Forming of Trust Level to the Nodes in the Distributed Computer Systems [Text]: Proc. of XI-th Internat. Conf. / V. Y. Mukhin, A. Y. Bidkov, T. V. Duc // Modern Problems of Radio Engineering, Telecommunications and Computer Science TCSET'2012. – Lviv, 2012. – P. 362.