F O U N D A T I O N S O F C O M P U T I N G A N D D E C I S I O N S C I E N C E S
Vol. 42 (2017) No. 4
ISSN 0867-6356
DOI: 10.1515/fcds-2017-0015 e-ISSN 2300-3405

Security Aspects in Resource Management Systems in Distributed Computing Environments

Marcin ADAMSKI*, Krzysztof KUROWSKI*, Marek MIKA†,
Wojciech PIĄTEK*, Jan WĘGLARZ†

Abstract. In many distributed computing systems, aspects related to security are getting more and more relevant. Security is ubiquitous and could not be treated as a separated problem or a challenge. In our opinion it should be considered in the context of resource management in distributed computing environments like Grids and Clouds, e.g. scheduled computations can be much delayed because of cyber-attacks, inefficient infrastructure or users valuable and sensitive data can be stolen even in the process of correct computation. To prevent such cases there is a need to introduce new evaluation metrics for resource management that will represent the level of security of computing resources and more broadly distributed computing infrastructures. In our approach, we have introduced a new metric called reputation, which simply determines the level of reliability of computing resources from the security perspective and could be taken into account during scheduling procedures. The new reputation metric is based on various relevant parameters regarding cyber-attacks (also energy attacks), administrative activities such as security updates, bug fixes and security patches. Moreover, we have conducted various computational experiments within the Grid Scheduling Simulator environment (GSSIM) inspired by real application scenarios. Finally, our experimental studies of new resource management approaches taking into account critical security aspects are also discussed in this paper.

Keywords: security, reputation, distributed systems, resource management.

* Poznan Supercomputing and Networking Center, Poznań, Poland, e-mail: {adamski, krzysztof.kurowski, piatek}@man.poznan.pl
† Institute of Computing Science, Poznan University of Technology, Poznań, Poland, e-mail: {marek.mika, jan.weglarz}@cs.put.poznan.pl

1. Introduction

There is no question that security aspects in distributed computing systems should be considered. As we present in this paper, they have a great impact on resource management processes, and consequently could change the overall performance. Today, an increasing number of remotely available computing resources in the form of Grid or Cloud environments with the easy access encourage many end-users to outsource their computations and data storage to remote, but often unknown physical locations. However, end-users intuitively pay a lot of their attention to security, reliability and availability of computing resources processing their data during computing jobs (tasks) submission and execution. In fact the basic trust relationship is established during authentication and authorization procedures while the end-user get the access to remote resources, but we believe that additional security aspects should be taken into account during later stages, in particular during tasks and resource management. In practice, many end-users tasks executed remotely could process important, sensitive or even critical data sets. In practice, even a highly efficient computing cluster or brokering system can be compromised in a second, e.g. scheduled computations can be much delayed or simply destroyed together with output data, if there is no management and control of IT security. Moreover, the trust relationship established over a certain period of time between a computing resources provider and end-users can be broken and it will be difficult to rebuild. Thus, we have performed many analyses of real workloads and log files collected in various distributed computing systems to find out if and how computing resources can be compromised. Based on the achieved results we have selected a set of key parameters that could represent the level of security of distributed computing environments. Additionally, we have defined a new metric called *system reputation*, which reflects in our opinion the level of reliability of computing resources from the security perspective. The proposed system reputation metric takes into account various relevant parameters regarding cyber-attacks, administrative activities such as security updates, bug fixes and security patches which can be extracted from historical accounting and monitoring data collected in many computing systems. The rest of this paper is organized as follows. In Section 2 we present briefly state-of-the-art and related works concerning reputation and trust relationship in distributed computing systems. In Section 3 and 4 we formulate relevant security metrics and key parameters. In section 5 and 6 we describe possible applications of our approach in relationship to not only computing, but also data-intensive environments. Section 7 presents basic components and features of our simulation environment - the GSSIM toolkit and various extensions required for our experiments together with different scheduling algorithms taking into account the new reputation metric. The simple experiment is presented in Section 8. Section 9 provides a summary of conducted research and presents our future work.

2. Related work

The reputation formula has been presented already in a few research papers over the last few years, e.g. authors in [13] have defined an interesting reputation formula which addresses the inherent unreliability and instability of worker nodes in large-scale donation based distributed infrastructures such as P2P and Grid systems. In the paper authors present adaptive

scheduling techniques that can mitigate this uncertainty and significantly outperform many other approaches. They present a new model in which reliability is not a binary but a real number taking into account the past and current system behaviour according to Formula 1 below.

$$r = \frac{completed\_tasks + 1}{all\_tasks + 2} \tag{1}$$

where:

r – reputation,
completed_tasks – number of tasks completed successfully,
all_tasks – number of all tasks in the system.
Authors use this relatively simple model to construct several reputation-based scheduling algorithms for efficient task allocations. The reputation is counted for each item (resource) of the computing infrastructure. One should note that at the starting point reputation is equal to ½. Each successfully completed task increases the reputation value in the considered system. In case of a task failure only *all_tasks* variable is increased, so each failed task does not change the value of *completed_tasks* variable and reputation value is still within the range (0,1).

3. New reputation formula

In this paper we propose a new resource management model which takes into account the new reputation formula, which does not depend on the number of tasks but on their execution time. The main criticism of the old reputation formula concerns a situation in which long-term tasks have the same meaning as short-term tasks but it is obvious that during one long-period task many short tasks could change the reputation of a certain resource to a great extent. Thus, in our approach, the resource on which many short tasks are computed has a greater reputation than the resource on which only one long-term task is computed. The proposed new formula for the reputation is as follows:

$$r_j = \frac{\sum_{i=1}^{N} t_i^s + 1}{\sum_{i=1}^{N} t_i^s + \sum_{i=1}^{M} t_i^f + \frac{T}{2}} \tag{2}$$

where:
r_j – the reputation of resource *j*,
t^s_i – the length of a task which ends with success *[in seconds]*,
t^f_i – the estimated length of a task which ends with failure *[in seconds]*,
i – task index,
N – the number of tasks which ends with success,

M – the number of tasks which ends with failure,
T – marked time section in which reputation is evaluated. The reputation always is computed in time interval whose length is known; which makes it possible to propose a new reputation value which is more adequate to length of time interval.
In general, the reputation computed in this way depends on a **task length**. It seems that the new reputation is a better choice for considered systems in which there are many long-term tasks, but on the other hand, there are a lot of short tasks. However, for systems in which a workload consists of only short-term or only long-term tasks the old reputation seems to be a better solution.
The new reputation formula can also be very interesting from the energy perspective. In case when a reputation is a measure of energy efficiency and security it is very important to run tasks on resources which are more secure from the energy perspective and it is more reliable that such task will be computed (each task which will not be computed wastes energy used for this task).
In the next section, we present the formulation of a new mathematical model using the aforementioned new reputation formula. Additionally, in the following sections we present an example of the application of our theoretical deliberations.

4. Problem definition

We consider an online resource management system to which tasks arrive continuously, and each one should be immediately served at the arrival time. In a certain moment, we consider only one task V_o with a known execution time t_j and a due time d_j. All tasks are independent and indivisible. If the execution time of task V_o exceeds its deadline, the task is stopped and it becomes a hostile task t^h.
There are $|R|$ resources, each r with certain reputations known in advance (the first reputation value is equal to ½, the next ones are computed based on the reputation formula) The reputation r_j of a resource *res* is recalculated every time a task is finished. We assume that all computing resources are identical (however, when it comes to hardware, the operation system can differ). A single resource is able to process more than one task at a time. We consider non-preemptive multitasking, i.e. any task being processed on a computing resource cannot be preempted until it is finished. In fact, in many existing resource management systems, multiple processes, also called tasks, can execute (i.e., run) on a single computing resource simultaneously and without interfering with each other. The multitasking is possible thanks to the use of a time slice technique, but typically this capability is implemented within the kernel of operating systems managing a limited number of computing resources and thus is not considered in our model. Additionally, we know all the previous allocations of tasks to resources at any time, thus, we are able to calculate the utilization of resources.
The main problem is to allocate all new tasks on available resources dynamically, in a way that completes all of these tasks by their deadline, and therefore, the overall reputation is maximized.
In general, we could formulate the considered problem as a multi-criteria resource management problem with the following two objectives: reputation and resource load. Then, in order to simplify the problem we assume that the resource load criterion is moved to constrains. Consequently, we define a threshold as the constraint which simply allow us to

check if a specific resource can be considered for the task execution. If the load on a resource is above the threshold the resource is excluded during the optimization procedure, otherwise the resource is considered.

The following notation is used for the problem formulation:

Indices and sets:

j resource index,
t_i execution time of task i,
d_i the deadline of task i,
R set of resources $\{res_1, ..., res_j\}$,
$|R|$ size of set of resources,
t^h hostile task h,
V_N new task,
$V_1 .. V_{N-1}$ currently executed tasks,
$|V^j|$ size of set of tasks computed on resource j.

Parameters:

r_j the reputation of resource j,
u_j resource load,
u^n_j predicted resource load,
$\underline{t}_N$ estimated time of counting task V_N,
T deadline for counting task,
T^H threshold of resource load, above this value new task will not be added to this resource,
a_j parameter with value from set {0,1}; 1 means that the task can be computed on resource j; 0 means that the task can't be computed on resource j.

Variables:

x_j decision variable with value from the set {0,1}; 1 means resource is selected; 0 means resource is not selected.

Objective function

$$BCR = \max_{j \in 1,..,|R|} (a_j * x_j * r_j) \tag{3}$$

We define a new objective function (BCR – Best Resource) selecting the best resource to compute task V_N, as the maximum value from the set of reputations multiplied by parameter a and multiplied by decision variable x. Since the decision variable x has a value of 1 only for one resource, the objective function will choose the resource, for which the reputation multiplied by the parameter a will be maximum.

If we have more than one maximal value of function BSR we choose the first resource from such a set.

Constraints

$$\overset{\wedge}{j \in 1,...,|R|} a_j = \begin{cases} 0 \quad when & u_j^n > T^H \\ & or \\ & for\ i = 1..|v^j| \quad \left(\frac{t_i}{1-u_j^n}\right) > T \\ 1 & in\ other\ cases \end{cases} \tag{4}$$

$$u_j^n = \frac{u_j * |v^j|}{|v^j| - 1}$$

$$\sum_{j=1}^{|R|} x_j = 1$$

We assume that the threshold value should be selected by estimating the best value based on the experiences from a longer period of time. It should be ensured that all of the tasks will be computed. For all tasks executed on the single processor we should be sure that in case of adding a new task, all tasks will be counted before the deadline. Thus, we estimate a predicted time of counting each task based on the estimated resource load. The estimated resource load is computed based on the last known resource load proportionally to the number of tasks.

Basic example

In the example below we assume that the time limit (**T**) for calculating one task is equal to 1 minute and the **threshold** value of resource load ($\mathbf{T^H}$) is equal to 74%. It means only res_1, res_2 and res_3 resources are chosen for the next step (the value of their load is lower or equal to the threshold). From these resources we exclude the res_1 resource because executing time is equal to 120s and it is two times more than T limit (deadline). In step 3 we choose the resource with the biggest reputation value res_3 (**r_3=0,4**).
In the practical scenario it means that the resource res_3 will service the next task (V_8) and its load value will be higher. So probably for the next step of the scheduling algorithm (V_9) res_3 will not be chosen because its load value will be higher than 75%.

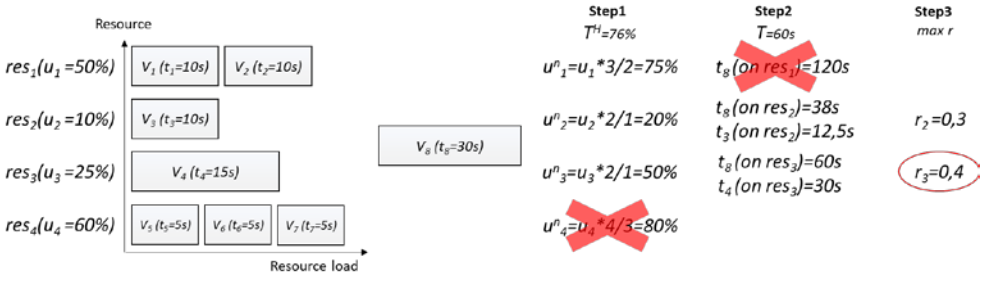

Figure 1. *Basic scheduling algorithm*

Scheduling algorithm

Let us now define a basic scheduling algorithm consisting of three main steps.
a). In the first step all resources with the load value lower or equal to the threshold are selected. If no resource is selected then the threshold's value should be changed to a greater value and step one should be repeated.
b). In the second step the estimated executing time for each task on a given resource is compared to the deadline. If the deadline is exceeded the resource is excluded from the scheduling process.
c). In the last step one resource with the highest reputation value is selected as the best solution from the whole set of resources collected in the previous two steps. The scheduling algorithm is relatively simple but it guarantees that all tasks are scheduled.

5. On-line web digital resources distribution scenario

One of the interesting resource management systems to use the new reputation concept is the on-line IT system for open educational resources distribution [11]. It is an advanced and complicated IT system, which has to serve even a few million web request operations per day. The main features of the considered system are presented below:

- It is an on-line dynamic system, all the request operations (tasks) should be immediately served by the system on available computing resources,
- it is possible to define a deadline for each request operation, e.g. an accepted response time,
- in case of a task failure, the task should be computed on an alternative resource from a set of computing resources,

There are three types of request operations:

- Static requests
  - getting image, css or static text
- Application requests
  - searching requests
  - context user requests
- Other requests
  - requests to generate a content of the platform pages

The overall architecture of the system is presented in Figure 2.

For every single TCP/IP request, the Linux virtual system (a front-end) randomly takes one of the available Varnish servers. The selected Varnish server classifies the request as one of the mentioned types and sends it further to a specific computing resource responsible for serving this type of requests. In this scenario, application requests are very attractive from the scheduling perspective as they depend on many third party subsystems, e.g. databases, authorization systems, and they are processed in much longer periods of time comparing to static requests and others. In fact the Varnish server has to schedule a specific application request onto the best computing resource in order to finish it as soon as possible. We propose to improve this scheduling procedure by applying our new scheduling procedure based on the new reputation formula presented in the previous section.

The algorithm may be used in such a way that in the first step a set of resources with load below the threshold can be selected and in the next step the algorithm finds the one with the biggest reputation from selected resources. As the presented IT system is currently setup on a critical infrastructure with a limited access to internal scheduling procedures, we have decided to take real log files to extract data and create real workloads together with a similar scheduling structure and characteristics within GSSIM. Consequently, we are able to create new experiments and verify our model as well as the scheduling algorithm as it will be presented in Section 7.

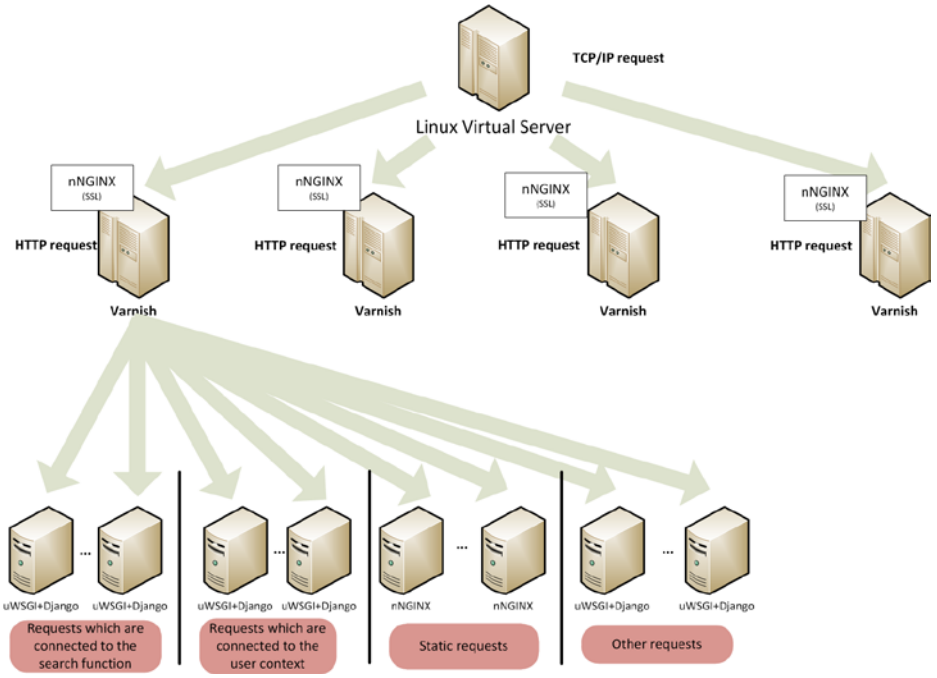

Figure 2. *The overall architectures of dynamic IT system in which a new reputation formula and scheduling procedure could be applied.*

6. Service orchestration - OGSA DAI scenario

The reputation can be also used for typical computing resources although such resources normally are grouped in computing clusters and rather should be treated as a single system with single reputation [1]. Typical infrastructure attacks are performed on whole system not for single resource (although it is possible to perform attack on a single resource too).
In distributed environment resources can be treated as services. In a real case one of the example can be OGSA-DAI services [10]. These services can be used to access different distributed data sources. There is the extended security mechanism designed for OGSA-DAI which covers authentication and authorization methods [2]. Additionally to this, the

reputation measure for each OGSA-DAI service could be proposed. Users could decide which service they want to use based on this measure. In common situation many OGSA-DAI services with similar functionality could compete with each other and a crucial value will be the reputation. The reputation, on the one hand, can be treated as a single value, and on the other hand as a part of multi-criteria value. Such value could define quality of service (QoS) measure. Also it is possible to use proposed (in section 7) scheduling algorithms to find the best OGSA DAI resource.
In the simple case the reputation formula from the equation (2) could be easily used for OGSA-DAI services. But it will be very interesting to extend this formula by others factors for example: the number of secure operation (authorization and authentication) could be object of our future studies.

7. Security and energy modelling extensions to GSSIM

Grid Scheduling Simulator (GSSIM) is a workload execution and simulation framework used by many researchers [4]. In a nutshell, GSSIM provides an automated framework for experimental studies of various resource management and scheduling policies in distributed computing systems [9]. GSSIM achieves this through a flexible design of architecture and interactions between scheduling components, a possibility of plugging scheduling algorithms into the simulated environment, modelling synthetic workloads and adopting real traces in popular formats and many other features. In general, input data in GSSIM consist of workload and resource descriptions. Workload provides information about tasks, their structure, resource requirements, relationships, time intervals etc. In GSSIM each task may have more complex structure with one or more tasks and preceding constraints between them, which enables constituting the whole workflow. Simulator allows one to adopt real workload (in most popular formats, like Standard Workload Format [6] or its extension Grid Workload Format [5]) or generating synthetic ones according to the statistical data provided by the user. In terms of resource modelling, resource description contains definitions of all resource providers available in an experiment. Each resource provider is described by a list of queues and information about available resources with specification of all single machines and their parameters, e.g. number of CPUs, memory, etc. Apart from that, GSSIM supports a definition of network topology connecting computing resources and afterwards it enables a detailed simulation of network traffic. GSSIM supports two-level scheduling architectures with two types of scheduling entities: Grid brokers and resource providers. Grid scheduler, is responsible for scheduling tasks to resources that belong to different administrative domains by the means of retrieving information about resources, submitting tasks, or creating reservations depending on specific settings and a type of considered scheduling problem. Local schedulers are responsible for managing resources that belong to a single administrative domain. They retrieve tasks and reservation requests from global schedulers and process them according to their own scheduling strategies. Offering those features, GSSIM can be configured to model a large scope of architectural patterns.
The output of each simulation consists of a number of statistics created to evaluate the simulated environment. They include both information about scheduled tasks (execution time, waiting time, completion time, etc.) and used resources (utilization, energy usage).

In order to run new computational experiments for the considered problem a set of new features were introduced to GSSIM. Conceptually, there are new items added at the global level - Global Broker and the resource level. New items render it possible to check a security policy at the resource level and propose new scheduling algorithms for the Global Broker (see Figure 3).

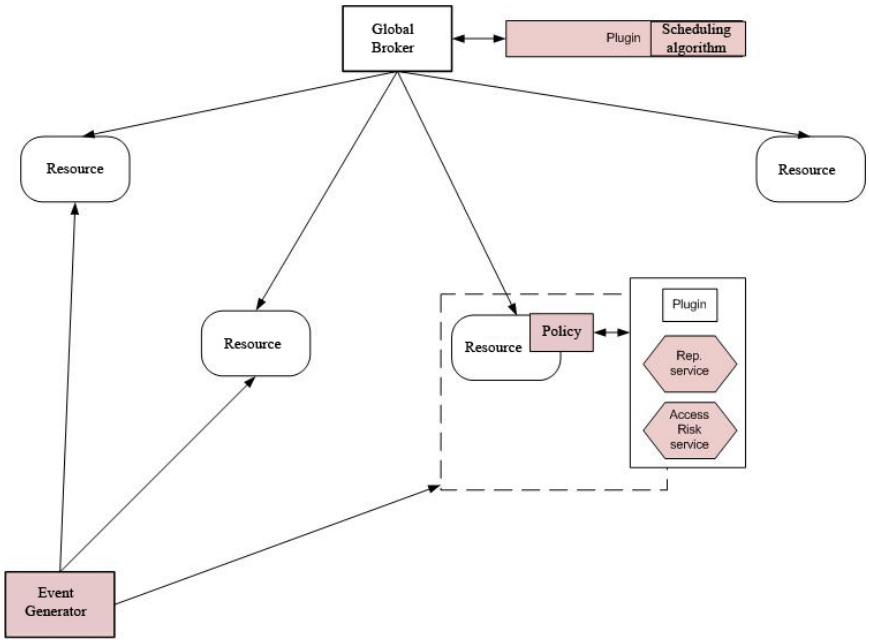

Figure 3. *GSSIM toolkit and security extensions*

A set of implemented extensions to the GSSIM toolkit:

- *Access & Risk service plugin* - this plugin (service) is storing all critical information about resources (e.g., if a resource is available or not). Based on the external information decision can be made. It is useful for simulating for example the DoS attack or Power Energy Attacks in this case Access Risk service returns false.
- *Reputation plugin* - the reputation plugin is used to describe resource ability to complete assigned tasks successfully. The reputation is counted based on equations proposed above. A number of completed tasks, attacks detected (also energy attacks) and performed software updates influence the reputation.
- *Event generator* – the new GSSIM item responsible for generating events like attacks or software updates for whole simulations infrastructure. Events are transmitted to all resources.

Additionally, on the Global Broker level a new set of scheduling algorithms have been successfully implemented and tested in the form of new GSSIM plugins:

- *GridSimpleReputation* – two steps algorithm which in the first step computes the reputation for every resources (based on the new reputation equation (2)). In the second step a resource with the biggest reputation is chosen to execute a task,
- *GridRoundRobin+Reputation* – multi steps algorithm which at first based on the standard Round Robin algorithm selects N resources and next for each selected resource computes reputation (based on the new reputation equation (2)). In the last step a resource with the biggest reputation is chosen to execute a task,
- *GridLoadBalancing+Reputation* – multi steps algorithm which at first based on the standard Load Balancing algorithm selects N resources and next for each selected resource counts reputation (based on the new reputation equation (2)). In the last step a resource with the biggest reputation is chosen to execute a task,
- *GridFCFSRoundRobinRandom* – multi steps algorithm which at first based on the standard Round Robin algorithm selects N resources and next from these resources randomly one item is chosen.
- *GridFCFSLoadBalancingRandom* – multi steps algorithm which at first based on the standard Load Balancing algorithm selects N resources and next from these resources randomly one item is chosen.

First three algorithms were developed for testing new reputation formula in real cases. Algorithm d. and e. were implemented to test the quality of other algorithms.
One of the most important functionalities covered by the "event generator" in GSSIM is the ability of simulating several events which are important from the scheduling perspective:

- cyber-attacks,
- power-attacks,
- bug fixes,
- thefts.

Cyber-Attacks are represented by a parametrized event which presents the power of the attack. Therefore, it is possible to model starting from weak attacks to mass attacks like for example the DDOS attack. The attack has influence on the availability of resources. Resources under the attack cannot be taken to the scheduling process. This is the reason why the reputation of such resources should be downgraded. This feature (attack event) makes it possible to simulate scheduler algorithms in an environment where resources are attacked.
The security patch represents an event which can be treated as the defense against the attack. It means that the resource after providing the security patch is less susceptible to attacks. It is possible to provide many security patches for a specific resource.
The **power attacks** ([15][17]) which are notably specific are very dangerous for computing infrastructures. Such an attack consists of attacking computing cluster or a part of computing center connected to one fuse. It is a typical solution that resources are not fully used (from the perspective of electricity) so it is possible to add more machines to the computing cluster (the power capping technology). A power attack means that all computers connected to one fuse are fully loaded and need more electric power than is provided for this electrical circuit. This results in the exclusion of the fuse and switching the connected infrastructure off.
The third aspect which was introduced to the GSSIM simulation environment is **theft**. Unfortunately, it is possible that computations are performed properly without any security problem but after executions end-users data could be stolen or compromised. It is a significant security accident which has to be mapped for a virtual equivalent. Thus, the power of theft variable was introduced to distinguish situations in which all data is stolen and

situations in which only a part of data is stolen. Unfortunately, typical log files from computing systems are published without the information about thefts. However, it seems that taking into account such events is important and should be considered by a scheduling algorithm. Thus, in the described models, both attacks and thefts affect the reputation. The reputation of a resource is reduced in the case of attacks or thefts. In other words, the scheduling strategy should avoid the use of resources where compromised data were located and prefer to use the resources with high reputations.
All the above-mentioned improvements allowed us to test different security aspects in GSSIM. Consequently, it should be relatively easy for researchers to repeat or create new scheduling algorithms and test them in the simulation framework. Finally, the new generic "event generator" function in the framework can be used in other scheduling simulations, and it is not at all limited to the considered security scheduling aspects.

8. Experiment

New functionalities described in the previous section were used to form a testing environment in GSSIM. The workload of 1500 tasks was created based on the database from the ACARM-ng system („Alert Correlation, Assessment and Reaction module - new generation" - http://www.acarm.wcss.wroc.pl/). The database contains a set of typical computing tasks and attacks against the infrastructure, the most popular was „SSH Brute-Force attack" (404 appearances), it contains also DDoA attacks (6 attacks). The considered computing infrastructure consists of 90 resources (each with 1 CPU). There is no data about the length of an attack so it was randomly generated from the interval of 2 to 3 hours. The average length of tasks execution was 4 hours (min 1,5 hour, max 60 hours). All tasks characteristics were known in advance.
In this experiment the new reputation formula was tested against the old one and a few relevant scheduling algorithms were compared to each other.
In the table below there are described results (in seconds) of scheduling 1500 tasks on the 90-resources infrastructure using algorithms proposed above (plus standard LoadBalancing and RoundRobin).

Table 1. *The results of the experiment*

	Old Reputation	**New Reputation**
LoadBalancing	995 373	971 997
LoadBalancingRandom	942 614	943 245
LoadBalancingReputation	969 680	967 000
RoundRobin	965 042	970 000
RoundRobinRandom	942 787	961 818
RoundRobinReputation	963 029	961 000
SimpleReputation	970 422	972 000

The obtained results by the GridLoadBalancing+Reputation algorithm in the form of Gantt chart are presented in Figure 4.

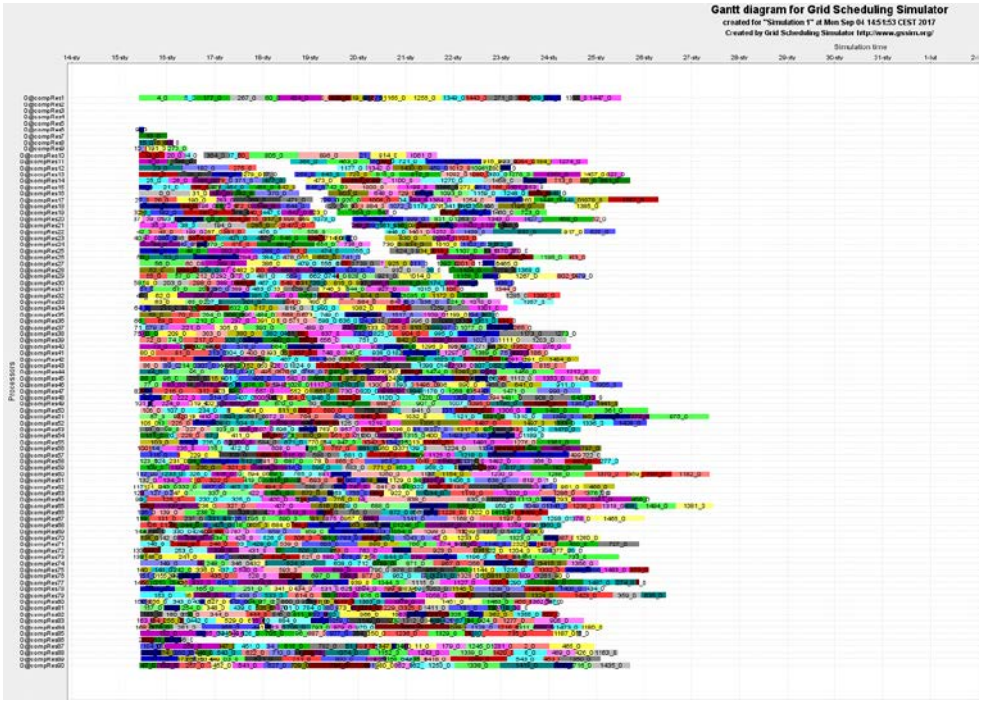

Figure 4. *The example Gantt chart generated by GSSIM after the simulation.*

In a set of performed computing experiments, it turned out that the GridLoadBalancing+Reputation was the most efficient algorithm in the light of new reputation objective. The worst solutions were generated by G*ridSimpleReputation* as all tasks were executed on a single resource with the biggest reputation value. Moreover, as expected the load balancing algorithms behaved better than the Round Robin algorithm. We have observed that in the considered workload, which contains many long-time tasks, the new reputation formula helps to select much better solutions than the old one. Therefore, the selection of old or new reputation formula for scheduling algorithms should be based on the initial workload analysis as the number of long vs. short tasks may effect the solutions quality.

9. Summary and future works

In this paper we presented initial studies of the security extensions to the GSSIM simulation framework and experimentally tested different resource management algorithms taking into account various new security aspects in distributed computing systems.
We have demonstrated that there is real need to address security aspects in resource and scheduling problems in computing systems as two example scenarios based on real IT systems, resource management and scheduling procedures have been analysed in this context. The proposed new reputation formula helps to determine the quality level of security for computing resources by taking into account two relevant types of security events: attacks and thefts, which could slow down the performance of computing resources or even exclude them during scheduling procedures. It is worth to emphasize that the new reputation does not depend on the number of completed tasks and the number of all tasks as it was defined on event-based security factors.
Our future works should address various new extensions to the presented reputation formula by taking new relevant security events that may effect the overall performance of scheduling algorithms in computing systems. We are planning also to collect and analyse a set of new log files and real workloads to check if and how detailed tasks characteristics may potentially impact the quality of solutions generated by scheduling algorithms.
Additionally, based on external studies, we will extend the reputation formula and our scheduling model to take into account energy consumption issues in the extended simulation framework called DCworms [15][16][17]. Thus, we plan to extend the reputation formula by a new energy efficient measure, which will define a relationship among an average number of hostile tasks and the energy usage. Finally, a new mathematical model will be proposed – the offline model for collecting a set of tasks in a batch for a certain period of time. Consequently, a new scheduling procedure will be possible to define and test in GSSIM or DCWorms where security and energy efficiency aspects will be taken simultaneously into account.

Acknowledgements

This work has been supported by the National Science Centre (NCN) in Poland under the MAESTRO grant Nr DEC-2013/08/A/ST6/00296.

References

[1] Aberer K. and Despotovic Z., Managing Trust in a Peer-2-Peer Information System, *in Proceedings of the Ninth International Conference on Information and Knowledge Management*, 2001.
[2] Adamski M., Kulczewski M., Kurowski K., Nabrzyski J., Hume A. C., Security and performance enhancements to OGSA-DAI for Grid data virtualization, *Concurrency and Computation Practice and Experience*, vol. 19, no.16, p. 2171-2182, November 2007.
[3] Alunkal B., Veljkovic I., von Laszewski G., and Amin K., Reputation-Based Grid Resource Selection, *Proc. Workshop Adaptive Grid Middleware (AGridM '03)*, 2003.
[4] Bąk S., Krystek M., Kurowski K., Oleksiak A., Piątek W., Węglarz J., GSSIM – a tool for distributed computing experiments, *Scientific Programming Journal*, vol. 19, no. 4, p. 231–251, 2011

[5] Delft University of Technology, Grid workloads archive, 2007, *[Online]. Available: http://gwa.ewi.tudelft.nl/*, [Accessed 2016]
[6] Feitelson D., Parallel workload archive, 2011, *[Online]. Available: http://www.cs.huji.ac.il/labs/parallel/workload/* [Accessed 2014].
[7] Damiani E., di Vimercati S.D.C., Paraboschi S., Samarati P., and Violante F., A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks, *Proc. Ninth ACM Conf. Computer and Comm. Security (CCS '02)*, p. 207-216, November 2002.
[8] Kamvar S., Schlosser M., and Garcia-Molina H., The EigenTrust Algorithm for Reputation Management in P2P Networks, *Proc. 12th Int'l World Wide Web Conf. (WWW '03)*, 2003.
[9] Kurowski K., Oleksiak A., Piątek W., Węglarz J., Impact of Urgent Computing on Resource Management Policies, Schedules and Resources Utilization, *In Procedia Computer Science*, 9, p. 1713-1722. Elsevier, 2012.
[10] OGSA-DAI platform, 2013, [Online]. Available : *http://www.ogsadai.org.uk*, [Accessed 2016]
[11] Open Educational Resources K12 platform, *[Online]. Available: http://www.epodreczniki.pl*, [Accessed 2015]
[12] Resnick P., Zeckhauser R., Friedman E., and Kuwabara K., Reputation Systems, *Communications of the ACM*, vol. 43, no. 12, p. 45–48, 2000.
[13] Sonnek J., Chandra A. and Weissman J., Adaptive Reputation-Based Scheduling on Unreliable Distributed Infrastructures, *IEEE Transactions on Parallel and Distributed Systems* 2007
[14] Sonnek J., Nathan M., Chandra A., and Weissman J., Reputation-Based Scheduling on Unreliable Distributed Infrastructures, *Technical Report 05-036, Dept. Computer Science and Eng. (CSE), Univ. of Minnesota*, November 2005.
[15] Wu Z., Xie M., and Wang H., Energy Attack on Server Systems, *in Proceedings of USENIX WOOT'11*
[16] Wu Z., Xie M. and Wang H., On Energy Security of Server Systems, *IEEE Transactions on Dependable and Secure Computing*, vol.9, no.6, p.865-876, November 2012
[17] Xu Z., Wang H., Xu Zichen, Wang X., Power Attack: An Increasing Threat to Data Centers, *In Proc. of NDSS*, vol. 14, 2014
[18] Zhao R. and Hwang K., PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing, *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 5, May 2007.

Received 18.10.2016, Accepted 22.10.2017