UDC 004.05

Pavlo Rehida, Artem Kaplunov,
Bohdan Ivanishchev, Oleksandr Honcharenko.

SECURITY ISSUES IN COLLECTIVE COMPUTING

This paper considers the issues of security of collective computing in a dynamic environment. Analysis of dynamic environment, where this type of computing is performing, is done. Metrics for assessing the security of the node are defined. These metrics can be used by the system to decide whether it is safe to continue working with a specific system node.

Key words: collective computing, computational trust,
Fig.: 2. Bibl: 4

Introduction. Nowadays, a lot of different computation devices are used. A huge part of them are connected to the network where they can transfer messages to each other. All of these devices have their own parameter that describes the computation power. According to low usage of this resources in big amount of devices, it is necessary to create a method to attract them for different purposes.

One of the key requirements of such computations is security of all system and computation security on each node. It is caused by the environment where all nodes works. Also, it is caused by that fact that system can’t trust all devices that performing computations. So, it is important to provide new security methods, that can be used as addition to existing methods.

Collective computing. It is a distributed system, that has several differences [1]. First of all, this system is heterogeneous and each connected node is a unique user of the system. It means, that all nodes are not equal to each other. Due to this, it is necessary to provide another algorithm for allocation tasks between nodes that will adapt to different computation power of node. All of these users provide their devices to take part in one of the commercial or scientific projects. One of the other differences is an environment where this systems works. This environment could be characterized as *dynamic* (Fig. 1)[2]. It means that while preparing the process of computation current system will have to take in account following restrictions: system's architecture is not defined, each connected node has different computation power, system can’t trust to any performed calculation on working node, system don't have stable connection with nodes (it means, that system can lose connection with worker in any period of time).

These restrictions does not allow to use classic methods, that provides appropriate level of security of the whole system. Also, this system have to be able to

work with a big amount of clients[3]. This will help to reach a good level of computation power. For this problem, it is necessary to use some mid-level servers. These servers will create a small systems to solve tasks in more efficient way. They have ability to control client's behavior and divide tasks between them.

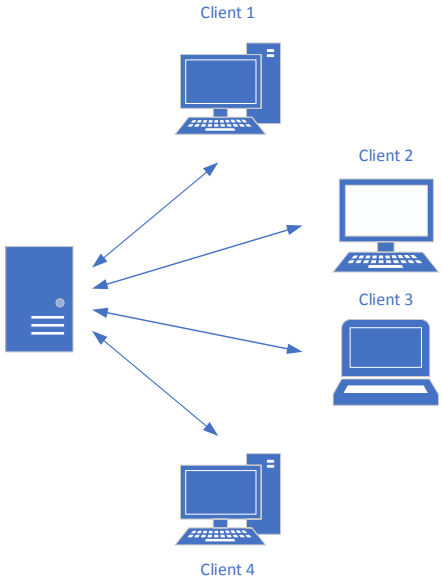

Fig 1. Dynamic environment.

Security. One of the good method to provide the high level of computation security in this environment based on recalculations. Also this method uses rating model for clients, that help to get know how many recomputations should be performed to get right result. System sends equal tasks to different clients and after their comparison. According to results rating of the client can be changed. As more correct answers client gives, the bigger rating he gets. Clients with bad rating can be banished from the system. To get better results we will talk about some methods that can work in addition to existing security system.

Security scheduler. All distributed systems uses schedulers to provide efficient way of dividing tasks between working nodes. This process have a lot of different strategies. All of them helps to pick right tasks from the queue. This work will show modified scheduler that aimed on security tasks and called Security Priority (SP) scheduler. It can be used as main scheduler on mid-level servers.

The main goal of scheduler is searching of the most appropriate tasks from working queue. We have to consider all main metrics for each tasks, to achieve this goal. For this scheduler they are: p_t – total priority, p_s – security priority, p_{s_max} – the highest possible security priority value, $t_{deadline}$– deadline time, and t_{calc} – calculate time. Here, one of the key parameter is p_s, this will help to focus on secure tasks. Next formula shows the usage of these parameters:

$$p_t = p_s + \frac{2 * p_{s\_max}}{t_{deadline} - t_{calc}}$$

According to this formula, you can see that tasks with small time to deadline will be placed higher in priority. Also, in situation when denominator less than 0, that means that task will not be performed in time. The priority of this task will be set to the minimum. This will help to save time on task that is guaranteed will not be executed. On next picture, comparison between EDF and SP schedulers is shown (Fig. 2).

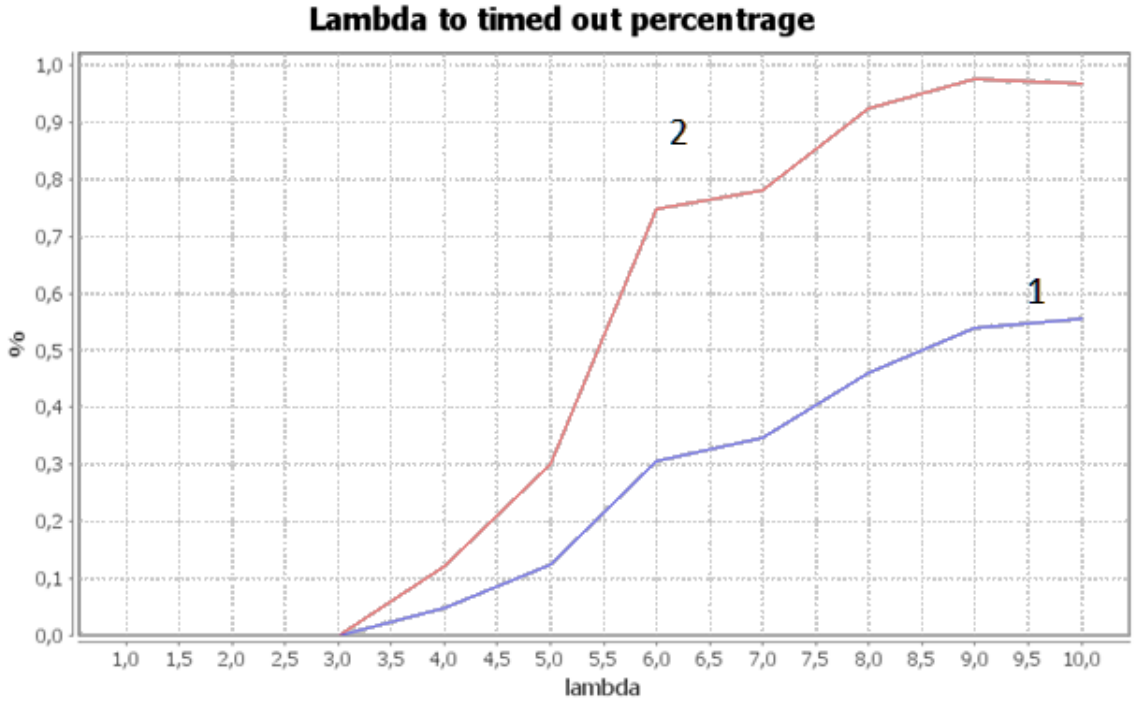

Fig. 2 Security priority scheduler.

This graph shows the dependence of probability of rejects for both schedulers on the intensity of the stream of new tasks. Fig. 2 shows that the dependence is significant bigger for modified scheduler than for EDF. SP effectively allocates tasks between working nodes.

Security metrics. To provide additional tool to increase the overall level of security in the system, the usage of new metrics suggested. We spoke earlier about rating system that can help to banish working node from the system. But, lets consider some new metrics that are important and can have influence on this process. The idea is to collect this data while working with computing node and give additional recommendations to the current security system about safety of each node. We define some of them: t_R – average time response, c_t – average count of computed tasks in specific period of time, t_w – average time of communication with the server, L – same location (%). Each system can work with different tasks, so it is recommended that this additional values can have system of weights,

that often used in neural networks [3]. This approach allows to tune system's settings to obtain appropriate result according to requirements.

To keep all information about users, system should have a profile for each working node. This data can be stored on mid-level servers. All of these servers keep information only about nodes with whom they work. Proposed additional tool have an algorithm, that consists of following steps: working nodes gets tasks to be computed, server creates profiles for new nodes, server collects analytical data while working with nodes, collected data can be used by existing security system.

Next step of developing and modifying this tool is to combine it with best practices of Artificial Intelligence (AI). First of all, machine learning algorithms is the best way to process data about all working nodes. The next step is to find out another not obvious but important metrics, that can indicate suspicious node. AI can help to find correlation between signs of banished node and new metrics that are testing right now.

Conclusion This paper is focused on security problems in collective computing. Some of key restrictions of dynamic environment was considered and analyzed. Security priority scheduler was proposed, comparison with other scheduler . Also, main metrics to analyze the working nodes was considered. Future work is considered: Proposed model of tool, that can be used as an addition to existing security system that collects node's data.

References

1) Talbot, D. (2001, April 1). Collective Computing. Retrieved from https://www.technologyreview.com/2001/05/01/235887/collective-computing/

2) Abbasi, H., Wolf, M., Schwan, K., Eisenhauer, G., & Hilton, A. (2004, September). Xchange: coupling parallel applications in a dynamic environment. In *2004 IEEE International Conference on Cluster Computing* (IEEE Cat. No. 04EX935) (pp. 471-480). IEEE.

3) Rencuzogullari, U., & Dwardadas, S. (2001). Dynamic adaptation to available resources for parallel computing in an autonomous network of workstations. ACM SIGPLAN Notices, 36(7), 72-81.

4) Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems* (pp. 1135-1143).

AUTHORS

Pavlo Rehida – assistant professor, Department of Computer Engineering, NationalTechnical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: pavel.regida@gmail.com

Artem Kaplunov – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: art.kaplunov@gmail.com

Bohdan Ivanishchev – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: art.kaplunov@gmail.com

Oleksandr Honcharenko – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”

E-mail: alexandr.ik97@ukr.net