Environment for Hybrid Simulation of Information Security Solutions for Grid and Cloud-systems

Valeriy Vasenin, Vladimir Roganov and Andrey Zenzinov
Institute of Mechanics, Lomonosov Moscow State University, Michurinsky pr. 1, Moscow, Russian Federation

Keywords: Grid Computing, Distributed Computer Systems, Information Security, Virtual Modeling, Simulation, Hybrid Modeling.

Abstract: Efficient and safe use of distributed information systems based on Grid and Cloud-computing technologies requires an assessment and optimization of their security level. For this purpose it seems appropriate to model the studied systems and their behavior under different working conditions. The paper presents an approach to modeling distributed systems in full-scale, simulation, virtual and analytical sense, and also describes the possibility of their simultaneous use in hybrid mode simulation. Such approach allows to investigate the behavior of distributed systems from different perspectives, taking into account features of the architecture, software, purposes of these systems and operation conditions. The paper also describes the mechanisms of the developed modeling software.

1 INTRODUCTION

In recent times, increasing attention is paid to software tools for security of large-scale distributed information systems. Despite the fact that there are a significant number of multi-level tools for Grid and Cloud-systems with such appointment, developing of security systems is very relevant and important research area. In order to provide the effective protection of such systems it is necessary to assess and adjust their security at different levels of their architectural implementation. This fact led to the problem: design and implement software tools for hybrid simulation of the Grid and Cloud-systems which provides unified model to present and simultaneously test several levels of the distributed system.

By the hybrid modeling approach we shall mean the conjunction of several different ways of modeling. This approach has already proven itself and is widely used in various fields of research and engineering which are closely related to complex systems, particularly in the space industry. It allows to study simultaneously a large number of different modes of behavior of a complex system without a real launch of a spacecraft. For Grid and Cloud-systems hybrid simulation we had chose Natural, Imitating, Virtual and Analytical modeling approaches. To stress the hybrid nature and wide range of simulation mechanisms the abbreviation NIVA (by first letters, means also Field in Russian) was used as name for consolidated modeling environment.

NIVA simulation environment does not require high qualification from users. The environment contains GUI-based editors for visual definition of configuration of designed distributed systems and, more important, their usage scenarios. It supports automated deployment of the configurations, interactive simulation mode combined with graphical visualization and batch control via command files, which was written on a standard interpreted language mixed with arbitrary decision logic. Usage scenarios of distributed systems are based on the template library of typical behavior patterns. With use of these patterns it is very easy to describe both normal and abnormal work flow of a distributed system, under conditions of the various attacks.

Thus, the environment allows to perform hybrid simulation of developed Grid/Cloud-systems in order to obtain their characteristics in terms of performance and information security, as well as support their optimization using multivariate non-interactive simulation with relatively small resource consumption.

2 EXISTING APPROACHES

The main idea to simulate distributed computing systems is not new and already has a long history. Start-

Vasenin V., Roganov V. and Zenzinov A..
Environment for Hybrid Simulation of Information Security Solutions for Grid and Cloud-systems .
DOI: 10.5220/0005105502550260
In *Proceedings of the 4th International Conference on Simulation and Modeling Methodologies, Technologies and Applications* (SIMULTECH-2014), pages 255-260
ISBN: 978-989-758-038-3

ing from epoch of grid concept a lot of simulation tools have been created, mainly based on discrete-event simulation, which, apparently, is the most versatile approach and has worked very well in the area of data communication networks. At present, the most famous and widely used specialized tools are SimGrid (Casanova et al., 2008), GridSim (Buyya and Murshed, 2002), CloudSim (Rodrigo N. Calheiros and Buyya, 2011), which are widely used for solving various tasks, such as optimization algorithms of key system components (Yaser Mansouri and Buyya, 2013). As an example of such kind of software developed in Russia the system developed in the Russian Institute for System Programming RAS (Grushin D., 2008) should be noted. In these environments, distributed system typically represented by a collection of objects belonging to its upper layer, that is, computational nodes, inter-node connections, etc.

For our purposes we chose SimGrid tool as base simulator. Its role would be discussed in the section 4. GridSim and CloudSim are written in Java, and SimGrid code is C-based, so it seems more convenient to perform large series of non-interactive experiments. SimGrid tool supports both Grid and Cloud-simulation, while GridSim and CloudSim are closely oriented on simulation of Grid and Cloud-systems, respectively, and we should use both of these two tools. We should also note SimGrid graphical configuration editor, implemented as an Eclipse plugin.

Yet another approach, based on virtual simulation is related to widespread virtualization support in hardware. In this case, the model simulates the instrumental level using the set of virtual machines interconnected by a virtual networks. Nowadays there are some papers about nested Cloud systems (Josef Spillner, 2012) but completely modeling of Grid/Cloud-technologies using virtual machines is poorly studied.

Of course, there are other approaches, such as analytical and full-scale modeling. If the simulation aim is to reach a large number of nodes (for example, to simulate large subset of the Internet), and is focused on the macro-level, then somebody can use the analytical methods such as Petri nets, etc. In contrast, the full-scale modeling can help to study such subtle phenomena in software as race condition vulnerabilities. Each of above approaches is good at its level, but none of them can fully and effectively cover the entire spectrum of possible interactions that take place in large distributed systems. Taking into account the above considerations, it seems appropriate to combine all of these approaches.

In addition, most of Grid and Cloud modeling software requires a programmer qualification to define and analyze the activities of interest. For this reason, the main priorities in the development of the NIVA environment were hybrid simulation support and operator comfort.

3 FUNCTIONALITY

Simulation environment NIVA for Grid / Cloud-systems supports mixed live, discrete-event, virtual and analytical modeling of distributed computing systems in both interactive and non-interactive modes, with an ability to visualize process and post-processing of results. The combination of different models allows a comprehensive study of large distributed systems, as well as to immerse them in a realistic context of the global network, namely:

- live (full-scale) simulation allows to evaluate the real computational efficiency of computational nodes and interconnection used on real hardware;
- virtual simulation helps to conduct the necessary experiments with the distribution, assess the compatibility of the software, and perform effective testing for known vulnerabilities;
- discrete-event simulation model is able to simulate the behavior of a significant number of Grid/Cloud nodes, as well as the consequences of possible attacks and the propagation dynamics of mortgage elements such as viruses;
- analytical modeling makes it easy to simulate very large Grid systems, such as significant part of the Internet, by taking into account the probabilities and intensity of corresponding dynamic system.

Operator's work consists of selecting/editing Grid/Cloud configuration, selecting/editing the usage scenario in the provided graphic editors, boot configuration before running the scenario and run the simulation with or without visualization, possibly followed by a detailed analysis of obtained results. Interactive modeling process can be stopped at any time moment by the operator (for example, for a detailed study of the state of the simulated system), run up the steps (in terms of virtual time) or prematurely terminated. There is also possibility to program an automatic pause/stopping trigger by certain conditions.

The result of the series of launches in batch mode is a report. Report helps to decide if the proposed tools and protection strategies are suitable, as well as make recommendations for further improvements.

Of course, any simulation environment have limitations. For instance, restrictions on the size of the

simulated systems dictated by the amount of available RAM and by the hardware platform performance. However, the advantages of the hybrid approach sometimes let get round this limitation by providing an opportunity for analytical modeling of individual large subsystems using empirical or statistical data. As a result, the hybrid approach allows to cover almost all cases considered Grid/Cloud-systems at the cost of accuracy.

4 HYBRID SIMULATION

Common part for various simulators in our case is a discrete-event simulation model as the most general. This approach eliminates the need to answer difficult question of the direct interaction of many diverse simulators. It requires minimum effort to create a "simulation representatives" for those entities that must interact directly, but are located in different simulators. As mentioned above we chose modern, actively developing SimGrid open-source framework as the basic simulator. It was extended with tools designed for interactive visualization, graphical scenario editor and scenario template library with definitions of typical activities in the Grid/Cloud-systems.

Hybrid modeling allows to use multiple simulators. Interaction of simulators can be organized in different ways. In our system we decide to do it on the basis of the discrete-event simulation model.

From the perspective of architecture it looks as follows.

1. Master discrete-event based simulator (for this case SimGrid) is the principal, and all entities (except those which are modeled analytically) have a representation as the nodes in the master simulation model. But some also have representation in the form of real processes inside virtual machines or real nodes in the case of virtual or full-scale modelling, respectively.
2. If a node is marked as running on an external (slave) simulator, then SimGrid only reassigns tasks to an external simulator and vice versa instead of the typical logic trigger actions. The external simulator communicates with SimGrid through a simple communication protocol.
3. External simulators receive complete system configuration with assigned actions. Therefore, they either interact with other nodes using "full-scale" way (i.e. conduct real DDoS attacks or send real malicious code), or just forward the message to SimGrid, if the destination node has only single imitation on the master side simulator.
4. Since the actions executed naturally instead of the simulation in the case of virtual or full-scale simulation, implementation support for these actions requires coupling logic. That is, when you receive a message with malicious code from a simulation model, full-scale simulator should transform it into a real malicious code. Therefore, in many cases it may be easier to perform independent modeling of these models and then compare the results to improve the level of confidence.

5 ESSENTIAL SOFTWARE COMPONENTS

Structurally software environment NIVA is a bunch of key components related to the following tasks:

- definition of configurations and usage scenarios of Grid/Cloud-systems;
- conversion of the configuration and scenario into low-level C program code to run it under main hybrid simulator;
- run the resulting program in one or more conditions;
- visualization and logging of significant events;
- post-processing of results and execution of subsequent operator commands (for interactive mode) or the control command file (for batch mode).

5.1 Graphical Configuration Editor Eclipse/SimGrid

Grid/Cloud-systems configuration editor is already present as addon for the SimGrid basic package and allows you to set and modify the sample configuration of Grid/Cloud-systems with use of IDE Eclipse. Eclipse framework is traditionally used as a wrapper in many modeling tools, and also includes tools for teamwork development (support for running with a network repository). Figures 1 and 2 show examples of the creation of system configurations complying "star" and "tree" topology, respectively. The user can perform most of actions by "drag and drop".

5.2 Graphical Scenario Editor

Scenario editor (determining activities in Grid/Cloud environment) has a simple graphical interface and allows you to define a usage scenario of a distributed system as a collection of parallel distributed activities. Activity hereinafter is a set of interacting processes in a distributed system that solve some dedi-

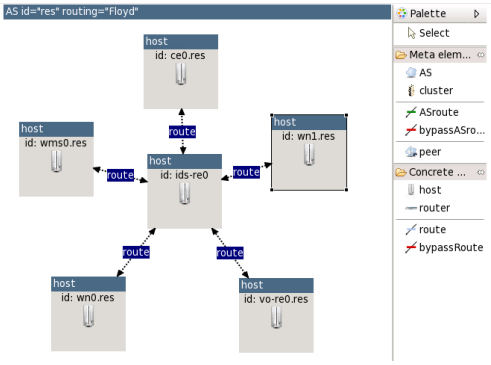

Figure 1: Eclipse/SimGrid configuration editor. Star topology.

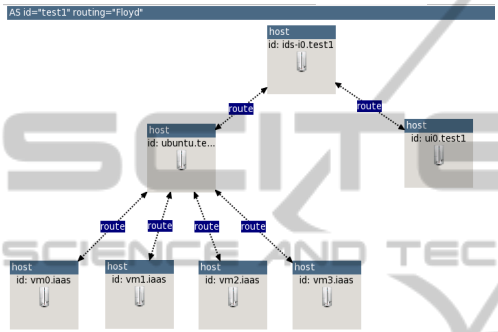

Figure 2: Eclipse/SimGrid configuration editor. Tree topology.

cated task together. As examples of activities you can consider user processes interacting during the execution of computational tasks, and also a set of attacker malicious actions during a planned attack.

Activity consists of actions. Some actions are preparatory, i.e. they performed before starting the simulation. This is a very important feature, which significantly simplifies the operator's work, because it is enough to modify (or, say, deactivate) some activity, slightly modifying the configuration of a distributed system before starting instead of frequent configuration changes.

On the Figure 3 you can see an example of "Virus" activity, which consists of "Virus" and "Bind" action templates. Each template can be adjusted in accordance to the requirements for activity and scenario.

Graphical scenario editor is implemented as a Java standalone application using a set of standard technologies — Swing, JavaBeans, and well-proven libraries. To simplify operator's actions many editing operations are performed by just dragging the objects using the mouse.

5.3 Visualization Server Scene

We have developed a server "Scene", that is a universal tool for displaying different states of the simulated system. SVG vector format, which is used in the Scene, is scalable and allows you to display complex graphic elements with the possibility of selective zoom (Fig. 4).

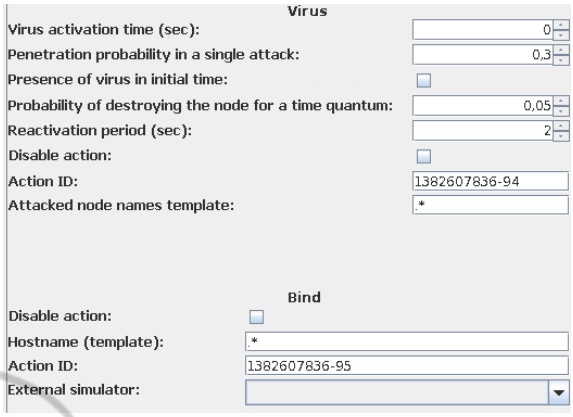

Figure 3: Activity parameters in scenario editor.

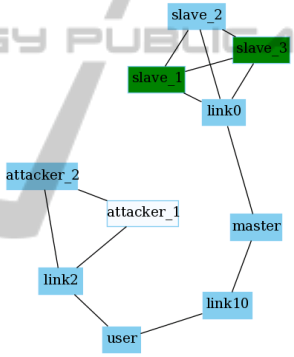

Figure 4: "Scene" visualization server.

"Scene" server supports the concept of "visual variables". For example, they can represent parameters which vary during the simulation and displays by the Scene directly with their changes (load/availability of nodes, node damages by the virus, etc.). Another feature is the ability to run Scene on a separate monitor (projector) as a server process with the ability to manage with multiple interacting parallel simulations.

"Scene" visualization server is implemented in Java and allows process group to display complex state of the modeled system. At the same time one of these processes declares itself as the master and sets the initial image (the map) passing URI of the corresponding SVG file. This SVG file may also include the logic for rendering changes, written in JavaScript. Each process involved in the simulation, whether the process simulation or a real process in Grid/Cloud environment can send requests to draw individual graphic signs (color change, text labels and so on),

as well as to receive information about changes in the "Scene", that may also be demanded in the more advanced interactive visualization.

5.4 Virtual Simulation System

Another distinctive part of the developed environment is a virtual simulator (Zenzinov, 2013). Virtual simulation is based on working with virtual machines and virtual networks. The management of these objects is based on the well proven cross-platform library libvirt. Deployment system is implemented as a set of programs written in Python. It performs the following functions:

- deployment of virtual machines on a given configuration;
- runs virtual machines with network service for performing tasks;
- when script execution is completed deployment system stops the virtual machine and does some cleanup (deletes temporary files).

Input data to generate a set of virtual machines consists of a common configuration for virtual machines, virtual network configuration and set of templates of disk images, which contains pre-installed operating system and the necessary software. Configuration files in this case are written in the JSON format and created automatically by the main configuration editor. Special network service installed on the virtual machine allows you to perform various tasks in the case of virtual or hybrid simulation. This approach provides the opportunity to use the real software in the process of modeling to carry out more detailed scenarios of a distributed system. For the use of any additional software you need to install it to the original templates of disk images.

5.5 Expanding the Environment Functionality

Since technologies used in the creation of distributed computing systems are continually evolving along with the tools and methods of their protection, the modeling environment for Grid/Cloud-systems shall be extensible. API of NIVA environment allows you to expand its functionality with relatively low resource and time consumption:

- the creation of new action patterns;
- extending the visualization server "Scene";
- in the case of hybrid simulation — implementing coupling simulation model with real programs launched during the simulation on the virtual or on real computers.

6 EXPERIMENTS

To illustrate the environment functions let us consider next three kinds of experiments.

6.1 Hybrid Simulation of Normal Grid-system Operation

In this experiment Grid-system consists of user node, master-node and slave-nodes. A typical scenario is the following: the user sends a request to master-node, which then distributes the work between slave-nodes. In this experiment slave-nodes were virtually modelled and the others were modelled in ordinary simulation mode.

As a typical computational task we've chosen the numerical calculation of $\int_0^{10} 3x^2 dx$ using Simpson's rule. On master-node this computational task represents as a small Python-script. Master-node sends it to virtual nodes via network service, and then it runs. Using virt-manager open source tool for virtual machine management you can follow task processing: its reception, launch and result.

6.2 Virus Attack Against the the IaaS Cloud-system

Second experiment is a simulation of virus attack against IaaS (Infrastructure-as-a-Service) Cloud-system from external infected network. Internal network of information system contains special intrusion detection system (IDS). Information system also contains node with IaaS infrastructure and simulated models of virtual machines, which are assumed to running on this node.

The experiment was conducted in two stages: with IDS turned on, with IDS turned off. In the first case IDS successfully blocked the virus propagation on the information system. In the other case we observed Cloud-system infection in accordance with the specified parameters of infection probability. Some of infected nodes have dead. Probability of failure was also specified by the parameter.

It should be noted that if IaaS-node fails, then its virtual machines also become unavailable and also terminate their activity. This situation corresponds to the real behavior of Cloud-systems.

Table 1: Summary results of virus attack

Configuration					
IaaS	PaaS	SaaS	Ring	Star	Line
Number of completed jobs					
23	33	51	15	47	5
Number of failed jobs					
10	5	0	13	1	18
Rate of successful jobs, %					
69.70	86.84	100	53.57	97.92	21.74
Number of dead nodes					
2	1	2	1	4	1

6.3 Non-interactive Launch a Series of Experiments

In this series of experiments we have investigated virus propagation and DDoS-attack against Grid/Cloud-systems with six different configurations. Three Cloud-systems: IaaS (Infrastructure-as-a-Service), PaaS (Platform-as-a-Service), SaaS (Software-as-a-Service), which differ in their architecture and mechanisms of job distribution. There are also Grid-systems with three different network topologies: ring, star and line. As in the previous paragraph there were two cases: IDS-on and IDS-off. This gives a total 24 different configurations.

When the execution of experiments is completed, the user receives a report which represents information about executed actions and a summary of results: the number of successful job queries, the number of unsuccessful queries, the number of dead nodes, and the overall success rate.

On the Table 1 you can see an example of this results in case of virus attack without IDS-protection. These numbers are completely dependent on input parameters and probabilities.

7 CONCLUSION

As a result of this work we have created a prototype of software environment for modeling of Grid/Cloud-systems which supports several simulation modes. In addition to widely used simulation approaches Virtual-based modelling allows us to assess the effectiveness of security systems, taking into account the features of real programs that can contain serious bugs. Combined with other simulation modes it gives us new unique possibilities with hybrid mode.

Our future work will be focused on improvement of our virtual simulator and its interaction with SimGrid. In general, hybrid modeling of distributed systems gives us a wide area for further research. There are a lot of tasks and questions related to the concept of hybrid simulation like improvement of the simulators interaction, search and investigation the specific effects resulting from this interaction etc. In addition, there are many problems in which modeling environment serves as a tool: estimation of security software for information systems, testing of special software for distributed computer systems, search for the optimal configurations to protect systems against specific attacks.

REFERENCES

Buyya, R. and Murshed, M. (2002). Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. In *Concurrency and Computation: Practice and Experience (CCPE)*, volume 14, pages 1175–1220. Wiley Press, USA.

Casanova, H., Legrand, A., and Quinson, M. (2008). Simgrid: a generic framework for large-scale distributed experiments. In *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, UKSIM '08, pages 126–131, Washington, DC, USA. IEEE Computer Society.

Grushin D., Kuzjurin N., P. A. S. A. (2008). Modeling grid behavior using workload data. In *The 3-rd International Conference "Distributed Computing and Grid-technologies in Science and Education"*.

Josef Spillner, Andrey Brito, F. B. A. S. (2012). Analysis of overhead and profitability in nested cloud environments. In *Cloud Computing and Communications (LATINCLOUD)*, pages 13–18, Washington, DC, USA. IEEE Computer Society.

Rodrigo N. Calheiros, Rajiv Ranjan, A. B. C. A. F. D. R. and Buyya, R. (2011). Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. In *Software: Practice and Experience*, volume 41, pages 0038–0644. Wiley Press, USA.

Yaser Mansouri, A. N. T. and Buyya, R. (2013). Brokering algorithms for optimizing the availability and cost of cloud storage services. In *Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science*.

Zenzinov, A. (2013). Automated deployment of virtualization-based research models of distributed computer systems. In *Proceedings of the 7th Spring/Summer Young Researchers Colloquium on Software Engineering (SYRCoSE 2013)*, pages 128–132. National Research Technical University Kazan, Russia: Kazan.