

Leveraging the Grid for the Autonomic Management of Complex Infrastructures

Silvio Salza

Dip. Informatica e Sistemistica,
Università di Roma La Sapienza
Via Salaria 113- 00133 Rome, Italy
salza@dis.uniroma1.it

Yuri Di Carlo, Flavio Lombardi, Roberto Puccinelli
Sistemi Informativi - CNR

P.le Aldo Moro, 7 - 00185 Rome, Italy
fax no.+390644236544

yuri.dicarlo@tin.it, flavio.lombardi@cnr.it, roberto.puccinelli@cnr.it

Abstract

Autonomic Management of complex IT infrastructures requires multiple different types of analysis to be performed on large data sets in order to promptly detect anomalies and attacks and take appropriate actions. The requirements in terms of computing power can be so high that new solutions must be devised in order to reduce reaction times. This paper describes how Grid technologies can be leveraged in the Autonomic Management of complex IT infrastructures, such as complex networks with tens of border firewalls. In particular, we show how an Autonomic Manager can benefit from the large computing power and storage space made available by the Grid. We designed the architecture of an Autonomic Manager which makes use of Grid resources to collect and store data and to perform analysis. We then implemented the core part of the system (LoGrid), which collects data in CBE standard format, processes them using Grid resources and feeds back the results in CBE format to the Autonomic Manager. We tested our implementation in a reference scenario and found that the use of the Grid for the Autonomic Management of complex IT infrastructure is feasible and convenient.

Keywords: Grid Computing and Applications, Autonomic Computing, Network Management

1 Introduction

IT infrastructure autonomic management [7] is, to a great extent, based on resource monitoring through log file analysis. Log file sizes can sum up to hundreds of megabytes or more. Oftentimes multiple different types of analysis must

be performed on large data sets in order to promptly detect and react to anomalies and attacks, as in the case of the autonomic management of multiple border firewalls [1]. In such cases, the requirements in terms of computing power can be so high that currently adopted technologies may not be adequate and new solutions must be devised or integrated, such as Grid computing[3]. This emerging technology, beside providing access to a large computing power, can also minimize the network traffic by shifting the computation close to the log sources.

This paper describes how Grid technologies can be leveraged in the autonomic management [12] of complex infrastructures. In particular we show how an Autonomic Manager can benefit from the large computing power and storage space made available by the Grid. Starting from the results of our previous research activity in the field of log analysis on the Grid [8], we designed a possible architecture of an Autonomic Manager which uses the resources of the Grid to collect and store log data and perform analysis. We then implemented the core part of the system (LoGrid), which collects data in CBE [10] standard format, processes them using Grid resources and feeds back the results in CBE format to the Autonomic Manager.

Currently, the maintenance of IT infrastructures heavily requires human intervention and not only the normal maintenance activities have increased, but also the type of faults and attacks and the probability of their occurrence. The most common tasks, security can be enforced by Intrusion Detection Systems and system managers generally make use of System and Network Managing Frameworks which allow complex infrastructures to be controlled from a single console. Nevertheless, when a problem occurs the system manager generally has to check what is happening, to devise

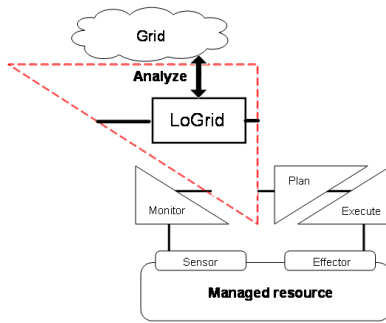


Figure 1. LoGrid and Autonomic Computing

a solution and implement it. This generally means reading large log files and trying to understand what the causes of the problem are. Autonomic Computing technologies aim to relieve the human operators from those tedious and often unfeasible tasks through the implementation of an automatic control loop which allows to deploy self-configuring, self-healing, self-optimizing and self-protecting infrastructures. This control loop is composed of:

- the system under control;
- the sensors, which gather data from the system and send them to the monitor;
- the monitor, which collects data from sensors, aggregates them and feeds them to the analyzer;
- the analyzer, which analyzes data captured by the sensors and detects problems;
- the planner, which plans adequate countermeasures in order to solve problems;
- the executor, which takes action by means of the effectors;
- the effectors, which act on the system according to the instructions issued by the executor.

The elements of the control loop generally make use of a knowledge base which keeps historical data and useful information. Monitoring of single devices is mainly achieved through log file analysis. In the following we briefly describe the reference technologies for the present work and describe the problem of log analysis in the Autonomic Management context. We then talk about our application, illustrating architecture, implementation and testing activity. Finally we comment on experimental results and draw conclusions.

2 Autonomic Computing: vision and tools

There are at least two slightly different visions of the Autonomic Computing. One is more focused on self-composing applications, the other on self-managing, self-healing, self-protecting systems. They are not antithetic but put a different stress on applications, on one hand, and infrastructures, on the other. Our job is clearly biased towards the latter and, in particular, towards complex IT infrastructures. In our vision the single systems composing the infrastructure should be seen as parts of a whole and, as such, managed. This increases the complexity of the analysis which needs to be performed on log fragments. In some cases problem detection cannot be achieved by just analyzing the log files of a single machine but requires the coordinated analysis of several information sources. This increases both the amount of data to be processed within a fixed time interval and the complexity of the computation. The requirements in terms of computing power can be very high and, in some cases, defeating even for high-end computers.

The second problem of complex IT infrastructure autonomic management is the extremely distributed nature of the information sources. Let's examine the case of a complex network containing tens (or hundreds) of border firewalls and routers. The log sources can be distributed on a wide geographic area and it may not be convenient to have a monolithic Autonomic Manager collecting data on a single storage system and performing the analysis. Such an architecture may cause a traffic overload on some network segments and also presents a clear bottleneck in the centralized processing. In those cases, data should be collected and pre-processed close to the sources, in a distributed fashion. Aggregated results may then be sent to a central system which performs a higher level analysis and coordinates problem resolution. This reduces the network traffic and also parallelizes the first-step computation. In this case the architecture is distributed with respect to both data and computing power. We have described a two-level computation and storage hierarchy but we can imagine, for very complex IT infrastructures, a hierarchy with more levels.

The last problem is log file format heterogeneity. Different applications and devices log events in different formats. It is very important, in order to simplify the system and facilitate coordinated analysis, to have a common format [14]. Having a single format means that the same analysis can be performed on different systems using the same software modules.

As a starting basis for our work we selected the IBM Autonomic Computing Toolkit [5], which provides the necessary tools to translate log files from any possible format to the Common Base Event format. This proved to be very useful to solve the format heterogeneity problem but has a

drawback and a caveat: the drawback is that the translation increases the log file size by an average factor of three; the caveat is that the translation should be performed online by the monitored resources or, where not possible, by the first level collectors. An off-line translation, performed after collecting a certain amount of data, introduces an often unacceptable delay.

3 Grid Computing

Grid Computing is an emerging technology which is being developed in order to provide seamless, coordinated and transparent access to distributed and heterogeneous resources (e.g.: computing elements, storage elements), even when they belong to different organizations. This allows to achieve considerable computing power and storage space even without having dedicated high-end devices. The Grid, according to the commonly accepted definition given by Ian Foster, Carl Kesselman and Steve Tueke [4], is a set of software technologies which allows resource sharing and coordinated resolution of problems. Thus, Grids allow institutions and users to achieve computational power and storage space that they could not normally afford. Let's see why an Autonomic Manager can benefit from the use of Grid resources. An autonomic manager can monitor a set of resources by processing their log files. In this scenario, the requirements in terms of computing power grows at least proportionally to the number of monitored resources, the complexity of the analysis and the number of records to be processed within a fixed time interval. Fortunately, in many cases the analysis can be parallelized by splitting the workload among an appropriate number of computing nodes. In those situations, the Autonomic Manager can increase its capability by using the Grid as a computing facility for log file analysis.

Currently, the de facto standard for Grid low level services is the Globus toolkit [2]. It provides the following items:

- Globus Security Infrastructure (GSI): the basic authentication-authorization system, based on the use of x509 certificates, which allows resources and users to be authenticated on a Grid using a unique identity;
- A Variety of Basic Grid Services which provide uniform interfaces to the most typical types of system elements, and in particular:
 - Computing / Processing Power (GRAM)
 - Data Management (GridFTP, DAI, RLS)
 - Monitoring/Discovery (MDS)
 - Authorization/Security (CAS)
- Developers APIs.

Starting from Globus 3.0, all the components have also an OGSA-compliant implementation (Open Grid Services Architecture), in order to make the Globus Toolkit interoperable with the emerging Web Services technology.

A certain number of high level middleware frameworks have been developed on top of Globus, in order to provide more complex services such as resource match-making and allocation, data replica management, etc.. For our work we used the LCG middleware, which includes components developed by the DataGrid and EGEE projects funded by European Union. The reason is twofold:

- In first place, it provides real resource transparency: the user does not need to specify the name of the machines which will process the data or where input data are stored, but just the high level requirements of the job (e.g.: the executable, the command line parameters, the number of CPUs) and logical names of the input files;
- secondly, this middleware allows easy interconnection with large multi-institutional testbeds in Europe, giving the opportunity of performing significant tests in a real geographically distributed environment.

An LCG testbed is made-up of a certain number of servers (at least one per type), which provide the high level services, and the resources the user needs to access (computing elements, storage elements, etc.). Here below we provide a brief description of each component of an LCG testbed.

High Level Services:

- The User Interface (UI) is the component that provides user access to all the DataGrid services (Job submission, Data Management, Information Management, etc.). The user interacts with the Grid via command line or using more advanced interfaces.
- The Resource Broker (RB) receives users requests from the UI and queries the Information Index to find suitable resources (ex.: a computing element which satisfies users requirements).
- The Information Index (II), which can reside on the same machine as the Resource Broker, keeps information about the available resources.
- The Replica Manager (RM) coordinates file replication across the testbed from one Storage Element to another. This is useful for data redundancy but also to move data closer to the machines which will perform computation.
- The Replica Catalog (RC), which can reside on the same machine as the Replica Manager, keeps information about file replicas. A logical file can be associated

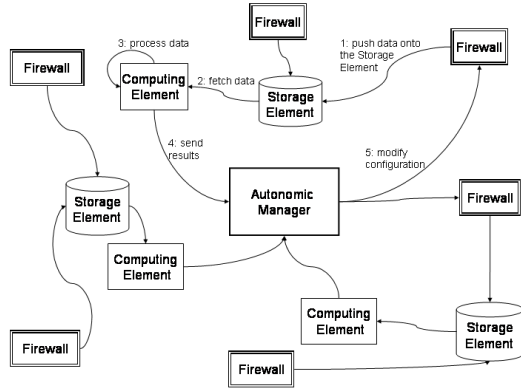


Figure 2. LoGrid architecture

to one or more physical files which are replicas of the same data. Thus a logical file name can refer to one or more physical file names.

Grid Resources:

- the Computing Element (CE) receives job requests from the Resource Broker and delivers them to the Worker Nodes, which will perform the real work. The Computing Element provides an interface to local batch queuing systems (e.g. PBS, LSF, ...). A Computing Element manages one or more Worker Nodes. A Worker Node can also be installed on the same machine as the Computing Element.
- The Worker Node (WN) pro-processes input data and produces output for a job.
- The Storage Element (SE) provides storage space to the testbed. It provides a uni-form interface to different Storage Systems.

4 LoGrid: Architecture

Figure 2 describes the architecture of an Autonomic Manager which uses Grid Resources for data collection and processing. For the sake of simplicity, some control signals have not been represented in the picture.

Border firewalls (the managed resources) push data onto the Grid storage resources. The storage resources write data onto local files and send notifications at fixed time intervals to Grid computing resources, which can reside on the same machines or on different ones. A reasonable time interval is 5 minutes, because shorter ones may lead to detecting false positives, and longer ones imply too slow reaction times. The computing resources process data and send the result to the Autonomic Manager. If problems are detected (e.g.:

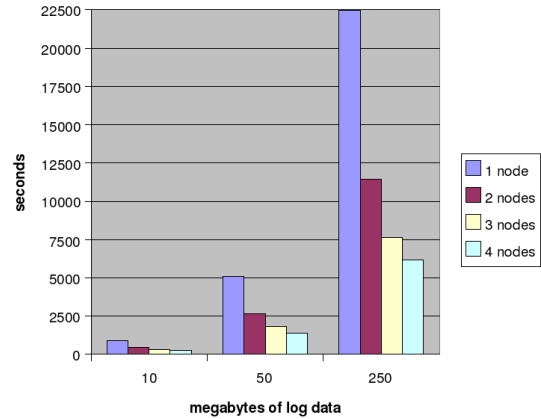


Figure 3. Analysis execution times

an attack), the Autonomic Manager acts on the firewall configuration in order to solve them.

In order to test the effectiveness of such an architecture, we decided to first implement the component which performs log analysis on CBE records using the Grid. The implementation is partly based on the results of a previous work [8]. We decided to use Java for portability reasons.

5 Testing activity

The main goal of the experimental activity is to show the effectiveness and the benefits of adopting the Grid for the autonomic management of complex infrastructures, such as large networks. To the best of our knowledge there is no other experimental result in literature regarding the application of the Grid to the autonomic management of border firewalls, whereas various degrees of integration between Grid and Autonomic computing have been recently proposed [6], [9], [11].

We applied two linear complexity algorithms to detect Distributed Denial of Service attacks. In particular, those algorithms consider the ratio and the difference [13] between the number of logged syn and fin packets per single destination IP address and compare them to given thresholds. We found that, for a given number of computing nodes, processing times did not change very much from one experiment to another.

We performed the tests with different log file sizes in order to verify the behaviour of the computing infrastructure with respect to the speed-up obtained from parallelization. The graph in fig. 3 shows how computing time is influenced by the total amount of log data. The results present a slightly sub-linear speed-up for each input data set. A considerable part of the computing load is due to CBE log file parsing.

Then, we considered the case of a network with 20 border firewalls, which can be considered a mid-size network. Using a standard logging level, a single firewall can produce about 100 KB of data in CBE format per minute. This means that the entire system produces roughly 10 megabytes of data every 5 minutes ($20 \times 100 \times 5 = 10000$). We processed two different 10-megabyte data sets using an increasing number of computing nodes (650Mhz Pentium IIIs with 256MB Ram each).

The following table shows the average times recorded during our tests:

	1 node	2 nodes	3 nodes	4 nodes
sec.	899.64	465.5	309.9	265.74

The following table shows the speed-up ratio due to parallelization:

1 vs 2 nodes	1.93
1 vs 3 nodes	2.90
1 vs 4 nodes	3.39

The results show that the speed-up achieved through parallelization is almost linear when the number of computing nodes is increased from 1 to 2 or from 1 to 3, but it is clearly sub-linear when we use 4 nodes. This seems to suggest that there is an overhead due to parallelization that is at least proportional to the number of nodes. Figure 4 shows the benefit of using a distributed computing infrastructure in the above-mentioned scenario. Without parallelization the computing time is three times the reference interval of 5 minutes, which defeats any attempt of analyzing data before the next five minute's worth of data is ready. Using 4 nodes the computing time is about 4 minutes and 25 seconds, which leaves 35 seconds to the Autonomic Manager to react to an attack before the next data set is ready.

In large networks the number of firewalls can sum up to 100 or more and more types of analysis must be performed on log data in order to detect different types of attacks and anomalies. This could render parallelization even more convenient than our experiments show.

6 Conclusion and further work

This paper describes how an Autonomic Manager can benefit from the large computing power and storage space made available by the Grid. We designed the architecture of an Autonomic Manager which uses the resources of the Grid to collect and store data and perform analysis. We then implemented the core part of the system (LoGrid), which collects data in CBE standard format, processes them using Grid resources and feeds back the results in CBE format to the Autonomic Manager. We tested our implementation in a reference scenario and found that the use of the Grid for

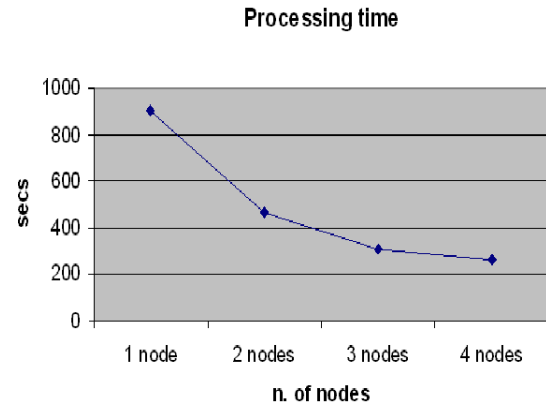


Figure 4. Analysis execution times for the given scenario

the Autonomic Management of complex IT infrastructure is feasible and convenient. On the other hand, the analysis of the results suggests the existence of an overhead which is at least proportional to the number of computing nodes. This is probably due to the Grid middleware. Further investigation is needed and we are working on the subject.

References

- [1] E. S. Al-Shaer and H. H. Hamed. Discovery of policy anomalies in distributed firewalls. In *INFOCOM*, 2004.
- [2] G. Alliance. Globus - <http://www.globus.org>.
- [3] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufman Publishers Inc., 1999.
- [4] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150:1–10, 2001.
- [5] B. Jacob, R. Lanyon-Hogg, D. K. Nadir, and A. F. Yassin. A practical guide to the ibm autonomic computing toolkit. IBM Redbooks, April 2004.
- [6] J. Kaufman, T. Lehman, G. Deen, and J. Thomas. Optimal-grid: Autonomic computing on the grid, 2003.
- [7] J. O. Kephart. Research challenges of autonomic computing. In *ICSE '05: Proceedings of the 27th international conference on Software engineering*, pages 15–22, New York, NY, USA, 2005. ACM Press.
- [8] F. Lombardi and R. Puccinelli. Jet-lag java heterogeneous log analysis on the grid: architecture, implementation and performance evaluation. In *Proceedings of Parallel and Distributed Computing and Networks (PDCN)*, Innsbruck, Austria. ACTA Press (Calgary), February 2005.
- [9] F. Martinelli, P. Mori, and A. Vaccarelli. Towards continuous usage control on grid computational services. In *ICAS/ICNS*, page 82, 2005.
- [10] D. Ogle, H. Kreger, A. Salahshour, and al. Canonical situation data format: The common base event v1.0.1, 2004. IBM Press.

- [11] A. Sajjad, H. Jameel, U. Kalim, S. Han, Y.-K. Lee, and S. Lee. Automagi - an autonomic middleware for enabling mobile access to grid infrastructure. In *ICAS/ICNS*, page 73, 2005.
- [12] M. Salehie and L. Tahvildari. Autonomic computing: emerging trends and open problems. In *DEAS '05: Proceedings of the 2005 workshop on Design and evolution of autonomic application software*, pages 1–7, New York, NY, USA, 2005. ACM Press.
- [13] H. Wang, D. Zhang, and K. Shin. Detecting syn flooding attacks, 2002. In *Proceedings of IEEE INFOCOM '2002*.
- [14] D. Worden and N. Chase. Using the generic log adapter with the log and trace analyzer, 2004. IBM DeveloperWorks.