Support Multi-Dimensional Trustworthiness for Grid Workflow

-Extended Abstract-

Elisa Bertino
Computer Sciences Dept. and CERIAS Purdue University
bertino@cerias.purdue.edu

Bruno Crispo
Computer Science Department Vrije Universiteit
crispo@cs.vu.nl

Pietro Mazzoleni
Dipartimento di Informatica e Comunicazione
University of Milan, Italy
mazzolen@dico.unimi.it

1 Introduction

Grid systems were initially developed for supporting scientific computations, in areas such as biotechnology and physics, and therefore their purpose was mainly to support computationally intensive tasks. Today, companies, users and researchers are looking at ways to use the grid approach for commercial uses and for applications in many different areas, ranging from the entertainment to the financial industry. The development of techniques and tools for coordinating the execution of complex tasks over grids, such as workflow management systems, as well as the development of specialized web services and of storage services, is simplifying the deployment of grid-based applications.

However, a limiting factor to the wide exploitation of the computational grid paradigm is represented by the untrusted environment in which computations are sometimes to be performed. Not only a grid system may encompass a variety of nodes very much heterogeneous with respect to trustworthy whereas others are not - but also the notion itself of trustworthiness is quite articulated. Trustworthiness may encompass several notions, like integrity, confidentiality, availability, privacy, reliability, to name some, and may be evaluated according to different criteria and mechanisms. Since we cannot expect that a single notion be suitable for all the possible applications and users in a grid environment, we need more flexible framework to assess and enforce trustworthiness which should allow users and applications to choose and select among various security requirements, for example to prioritize availability rather the confidentiality. It is also important to note that when a grid application is structured as a workflow process [2, 3], each of the composing steps may have different requirements with respect to trust.

When dealing with Grid trustworthiness it is useful to distinguish between two main notions of trustworthiness, namely Data trustworthiness and Computational trustworthiness.
Data trustworthiness refers to the trustworthiness of the hosts storing data needed as input for a task or produced as output of a computation. For example, a user might prefer a workflow task to store data into a node that guarantees high availability, even if the node cannot perform encryption and thus cannot provide confidentiality. Similarly, another user might wish to specify critical results of a computation being stored only into nodes which guarantee a high level of confidentiality even if not always available. Computation trustworthiness refers to the trustworthiness of the hosts which accomplishes the computations. For example, a user might wish his/her tasks to be executed only at nodes which guarantee a high level of confidentiality or in which the availability of the service for the entire computation is guaranteed.

In case of grid workflows, composed by several activities to be executed sequentially or in parallel by autonomous hosts, it is likely that security requirements of these activities about nodes storing data and those executing the computations using such data do not match. Thus, to complete a workflow, information needs to migrate or being replicated into different hosts each satisfying a different subset of the trust requirements. In such context, it is therefore important to have a comprehensive model dealing with data and computational trustworthiness at once. Figure 1 shows the main phases of executing a grid workflow. Computational trustworthiness needs to be evaluated when the scheduler selects the nodes among the ones available to execute a task (step 1 in Figure 1). Data Trustworthiness needs instead to be considered at steps 2 and 3 when the selected node loads input from the Grid or saves some results across the grid.

![Figure 1: Grid Workflow execution phase](image)

2 Contribution

In this paper, we elaborate on some of the concepts underlying such framework and point out relevant open research directions. In particular, our framework is based on the following key concepts:
1. An extensible set of grid node security properties (integrity, confidentiality, availability, etc.) - we assume that there is no specific single criteria. Rather there could be several security properties which are fixed by the user/applications and/or by the administrators of the grid. Security properties may be dynamically added and removed. Associated with each security property there will be a rating/label mechanism organized according to a partial order, that is, a lattice.

2. Flexible evaluation methodology - we do not assume that each grid node is rated according to all possible security properties. Rather we assume that each grid node may be rated according to an arbitrary subset of these properties or, even, not rated at all. Also, in our approach we assume that each rating criteria has associated one or more services allowing one to rate a given grid node with respect to the given security property. We assume both self-rating, according to which a node voluntarily rates itself for a chosen set of security properties, and third-party rating, according to which the rating is performed by entities different from the rated node. The user is completely free to select which security property to evaluate and which evaluation services rely at any time accordingly to his/her trust requirements. He can even implements his own evaluation service.

3. Trust-annotated workflow specification - in order to support the specification of articulated trust requirements for grid computations, we assume that for each task of the workflow, the user may specify required conditions for one or more security property and also specify the approach used to evaluated those (i.e., subjective rather than using third part evaluation services, etc.).

4. A scheduling algorithm with a set of relaxation strategies - the scheduling algorithm is in charge of devising the set of nodes where the various workflow tasks are executed taking into consideration, among the others, the trust-annotated specification. Since depending on the status of the grid and on the specification, it may not be always possible to find an execution schedule, strategies are provided in order to relax some of the constraints and/or to modify the workflow.

3 Methodology

The workflow model we use is based on the language proposed in [1] where a workflow basically consists on a set of tasks, partially ordered according to task dependencies.

Workflow tasks are defined using a tuple \((Act, Host, I, O)\) in which \(Act\) represents the list of activities to be executed, \(Host\) the node in which the task has to be processed, and \(I\) and \(O\) represent Input and Output data respectively. A workflow dependency describes the precedence order among tasks and the conditions when a task can be executed. A dependency has the form \(t_i \xrightarrow{x} t_j\) and states that the task \(t_j\) can start after (or along with) task \(t_i\) when the condition \(x\) is verified. \(x\) represents the dependency type and it is defined as a logical expression which specifies the conditions under which a task can be executed. The use of logical clauses to express dependencies makes the language flexible and adapt for being used into a grid environment as well as for being mapped into other languages.
In this paper we propose two enhancements to adapt such model to the grid domain. First we extend the model to support workflow specifications in which the execution host for some of the tasks is not known in advance. Second, we define an extended model allowing one to specify the source of input data for the tasks. We support three different types of source: (1) manual input by the user, (2) data load from the set of data produced by a previous task of the workflow, (3) data already available across the grid independently from the workflow. Similar specifications apply to output data as well.

Those extensions are the foundations onto which we apply the multi-dimensional trustworthiness criteria. We formalize the notion of lattices, usable by users to select multiple trust criteria. Lattices can be applied to both the \textit{host} (for Computational trustworthiness) and the \textit{I} and \textit{O} (for Data trustworthiness) of each task of the workflow. A host will be selected by the scheduler for executing a task (as well as for storing information) only if it satisfies the security conditions set by the user on behalf of whom the workflow is executed. However, we also deal with cases in which such conditions cannot be satisfied. As example, consider a scenario in which a user specifies that the input data for a task \textit{t_i} should be loaded from nodes whose lattices has a level of confidentiality sets at least as "high". If the input data is available only from a node with confidentiality="medium", an inconsistency arises. To solve such inconsistency, an immediate solution could be migrating input to another node matching the condition. However, the new node storing data should match all the security conditions by the task that generate the data (e.g., store data only into nodes which availability="high"). In case of several tasks sharing the same data, there might problems in planning a suitable assignment without storing information only onto grid nodes having lattices with higher values for all the possible criteria. Moreover, migrating data from one host to another is a costly process which might negate the advantage gained by using a Grid.

In this paper, we analyze the above issue as well as other inconsistencies that can arise in a workflow executed in a grid environment. We propose algorithms which take into consideration the trust requirements of all workflow tasks when deciding which node should store data as well as perform a computation. In our work, we do not have trust as stand-alone component for grid workflows, instead we develop a solution which takes into consideration other elements such as keeping minimal the number of data-migrations and balancing workload distribution among grid nodes having different security properties.

References

