Virtualisation Technologies – a new model to implementing Grids

Marcus Hardt*
Dr. Rüdiger Berlich, Dr. Marcel Kunze
Institute for Scientific Computing
Hermann-von-Helmholtz-Platz 1
76344 Eggenstein-Leopoldshafen
Germany
{berlich, kunze, hardt}@iwr.fzk.de

Abstract

One problem in distributed computing is bringing together application developers and resource providers to ensure that applications work well on the resources provided. A layer of abstraction between resources and applications provides new possibilities in designing Grid solutions.

This paper describes the virtualisation environment Xen (developed at the University of Cambridge / UK) and its fields of application, with particular emphasis on Grid computing. This includes the description of our grid-training environment implemented for a local Grid computing school on Xen and use cases that provide a solution to the problem that different Linux distributions are required by different grid users.

Deployment of large-scale distributed applications is a very difficult business. This is due to the large amount of parameters required to fully describe a computing system. These parameters include the architecture, the choice of operating system, the chosen OS distribution and the version being used. Even the compile-time settings used for creating certain libraries might be required to describe a system. In some applications it was found that even different settings for the compilation of the mathematical library `libm` can cause changes in the final results of calculations. In consequence high energy physics (HEP) experiments, such as D0 and CDF, have prepared certification toolkits which have to be run on new resources before they can be added to the experiments' resource pool. These certification toolkits cover the major problems that experiment software typically encounters, including to ensure that the experiment software will install properly on the remote machine.

The easiest solution for this problem would be to install standardised software on standardised hardware for all resources in the Grid. This contradicts one of the key design principles of the Grid, to support heterogeneous environments. Furthermore this is unfeasible in a fast-moving, global environment with dynamics due to ongoing development. In the European projects Data Grid (EDG) and CrossGrid (CG) this approach was taken as far as the installed software is concerned, because the main focus of both projects was the development of middleware and early adaptors' application software. However, this tied a considerable amount of manpower to system administration tasks.

Distributed middleware development is certainly easier if common assumptions on the installed resources can be made. However, this is not a generic approach.

Related to this are the conflicts between system administrators, middleware and application developers. All use different systems to maintain or develop their applications. This means the application developers own the systems they develop on, thus being able to tweak them to their needs. This results in different, often incompatible, assumptions being made on the resources where the software will be used. In a Grid environment their applications are supposed to work on all kinds of resources administered by a multitude of different administrators with typically different policies and restrictions. These applications are designed to use Grid middleware that is usually more integrated in the view of the system administrators.

In other words, there is currently no common understanding of how each node for distributed computing should be designed. A virtualisation layer of abstraction that gives each party enough freedom should be introduced. Two possible solutions, Xen and UML will be discussed.