International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:7, 2008

An Off-the-Shelf Scheme for Dependable Grid Systems Using Virtualization

Toshinori Takabatake

***Abstract*—**Recently, grid computing has been widely focused on the science, industry, and business fields, which are required a vast amount of computing. Grid computing is to provide the environment that many nodes (i.e., many computers) are connected with each other through a local/global network and it is available for many users. In the environment, to achieve data processing among nodes for any applications, each node executes mutual authentication by using certificates which published from the Certificate Authority (for short, CA). However, if a failure or fault has occurred in the CA, any new certificates cannot be published from the CA. As a result, a new node cannot participate in the gird environment. In this paper, an off-the-shelf scheme for dependable grid systems using virtualization techniques is proposed and its implementation is verified. The proposed approach using the virtualization techniques is to restart an application, e.g., the CA, if it has failed. The system can tolerate a failure or fault if it has occurred in the CA. Since the proposed scheme is implemented at the application level easily, the cost of its implementation by the system builder hardly takes compared it with other methods. Simulation results show that the CA in the system can recover from its failure or fault.

***Keywords*—**grid computing, restarting application, certificate authority, virtualization, dependability.

I. Introduction

Recently, grid computing has been widely focused on the science, industry, and business fields, which are required a vast amount of computing [1]–[3]. Grid computing is to provide an environment such as parallel computers or supercomputers, which many nodes (i.e., many computers) are connected with each other through a local/global network and it is available for many users. Although the environment of the grid computing is similar to that of a cluster computing, the cluster computing is difficult to connect with heterogeneous-typed computers each other. On the other hand, the grid computing can easily make the connection with the computers.

In addition as an example, to achieve the data processing among nodes in the grid computing, each node executes mutual authentication between the nodes, e.g., clients/servers, by using certificates which published from Certificate Authority (for short, CA). However, if a failure or fault has occurred in the CA, any new certificates cannot be published from the CA. As a result, a new node cannot participate in the gird environment. Furthermore, when the grid environment is recovering by building a new CA, the process of recovering can be consumed time in proportion to the system-scale.

To overcome the problem mentioned above, many methods for fault tolerance of grid resources have been developed [7]–[21]. These methods are as follows: fault detection and fault recovery (e.g., checkpointing and process migration, resource replication, etc.). Especially, one of the technologies is that of the "virtualization" [13]–[21]. Generally, virtualization is the technology to enable a single computer to support multiple operating systems simultaneously by virtual hardware interfaces [4]–[6], such as a host OS typed, a hypervisor typed virtual machine, and a virtual environment. The feature of the virtualization is to facilitate to build various execution environments by different operating systems through the virtual hardware interface. This makes it possible to coexist with various operating systems on the single computer, because the virtual hardware interfaces can absorb the variation of the different hardware [5]. Thus, the same execution environment can build on the different hardware physically.

In the grid environments, virtualization techniques for grid systems have been studied in [13]–[21]. Trusted computing is performed by a virtual machine-based platform for achieving a tamper-resistant hardware platform [13]. Virtual machines are provided and managed for the grid computing [14], which is particular to virtual machines being copied to provide the same execution environments across the grid environment. A distributed file system for virtual machines is supported in grid computing [15]. In addition, to provide security, a virtual hosting system through lightweight user-level virtualization is implemented [17]. This hosting method allows isolation, such as a storage, network, and process space. A virtual storage system is implemented by virtual organization repository [16]. Besides, for application, virtualization techniques are used in a university grid center [18]. On the other hand, for fault tolerance, a protocol of gird migration system by network storage has been studied and implemented in [20]. A job management system for detecting a process failure has also studied in [21]. These techniques have their drawbacks such that the cost of their implementations may take and also may have their complexity.

In this paper, concerning about the virtualization which can build the same execution environment on different hardware platforms, an off-the-shelf scheme for dependable grid systems using virtualization techniques is proposed and is implementation is verified. The proposed method is a way of restarting an application, e.g., the CA, which has been down because of its failure or fault. It is also verified whether the time of recovering the system to be reduced or not. The proposed method is that the CA running a platform (an operating system) on a virtual environment using the virtualization techniques is migrated on a different platform by some configuration files. The configuration files are copied on the different platform by using a backup tool periodically. Then, by using a surveillance program, i.e., a watchdog program, the CA is periodically confirmed whether it be down

Toshinori Takabatake is with the Faculty of Engineering, Shonan Institute of Technology, Fujisawa, email: toshi@info.shonan-it.ac.jp

or not. In this way, if the CA has become down because of a failure or fault, a virtual environment (virtual machine) is invoked from the configuration files, then the CA is recovered automatically. Since the proposed scheme is implemented at the application level easily, the cost of its implementation by the system builder hardly takes compared it with other methods. Simulation results show that the CA in the system can recover from its failure or fault.

The rest of this paper is organized as follows: In Section II, we provide preliminaries for this paper. Section III presents the proposed dependable grid systems. Section IV evaluates the proposed system. Section V discusses the proposed system. Finally, Section VI summarizes and concludes this paper.

II. Preliminaries

In this section, grid computing, certificate authority for fault-tolerance, and virtualization are described briefly.

A. Grid Computing

Grid computing is to provide that the mechanism of connecting each of distributed information resources (e.g., computers, databases, observation equipments, etc.) in a remote place through a network and that many users can share the connected information resources [1],[2]. The grid computing is classified into three types: computing grid, data grid, and service grid. Generally, these grids can yield high performance of information processing as parallel computers or supercomputers virtually.

Since low-cost computers are preferable to build the system of the grid computing, many general-purpose computers, which are composed of cheap commercial-off-the-shelf components, are used for the system. Thus, the system by using many such these computers is able to be improved the system performance even if those computers have used ones. On the other hand, the system being built by the low-cost computers is prone to occurring a fault or failure. As a result, the system reliability is reduced.

For the grid computing, a middleware is needed since it absorbs the difference in the specifications for each computer, and provides information for applications. There is the globus alliance which has developing such a middleware called the Globus Toolkit. This toolkit is going to be the standard to build the grid system [3].

B. Certificate Authority

A certificate authority is an entity which trusted by one or more users to create and assign public key certificates and be responsible for them during their whole lifetime. The role of the CA is very important in the grid environment. As described in the previous section, the grid environment is provided through some networks which are available for many users. Thus, security is required for the users. There are mainly two methods for the security in the grid system. One is mutual authentication using certificates by PKI and X.509 based on Certificate Authority (CA, for short) [1]–[3]. The certificates make use of SSL protocol and WS security. The other is proxy certificates which are the extension of X.509 (cf. RFC 3820). Delegation is implemented by the proxy certificates. Concerning to the CA, there are My Proxy and Grid Portal, and so on [1]–[3].

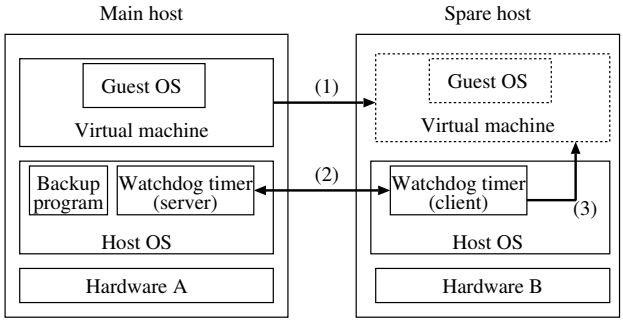

Fig. 1. Proposed off-the-shelf scheme of backup and recovery using virtualization.

By the mutual authentication, the server/client verifies each other, when the client throws a job into the server. Then, the job starts in the server. The necessary condition in the authentication is that the server/client has a certificate which proves for them. In addition, the certification has signed by the CA [1]–[3].

If a fault or failure has occurred in the CA by any causes, the CA cannot publish the certification. From this point, a new node (client or server) is not able to participate in the grid environment. As a result, the system performance, e.g., scalability and availability, is declined due to the faulty CA. Therefore, the CA should have its fault tolerant property.

C. Virtualization

Virtualization is a technique which can build any execution environments virtually on a computer. Many virtualization techniques use a virtual execution environment which is called a virtual machine [4]–[6]. Conventionally, more than one operating system cannot coexist in the hardware originally. However, the virtual machine makes any operating systems on single hardware running simultaneously. The virtualization was firstly used in the IBM mainframe computer [5]. Recently, virtualization has become noticed by the free software, e.g., Microsoft Virtual PC and VMware Player, due to an advanced high performance hardware [4]–[6].

III. Off-the-Shelf Dependable Grid System

In this section, an overview of the proposed methods in implementing a grid dependable system; the methods such as backup, failure/fault detection, and recovery are presented, respectively.

A. Overview of Proposed Methods

The proposed approach is to make a backup of the environment with the files, which are described the configuration of the system information. In this paper, the CA is used as one

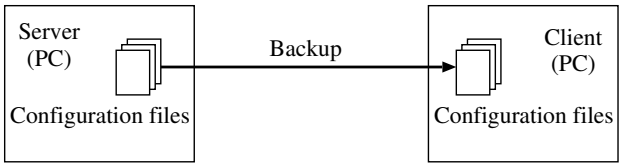

Fig. 2. Backup in the proposed method by virtualization.

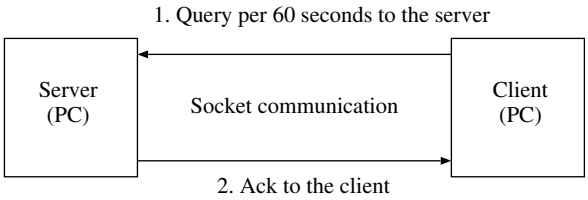

Fig. 3. Fault detection in the proposed method.

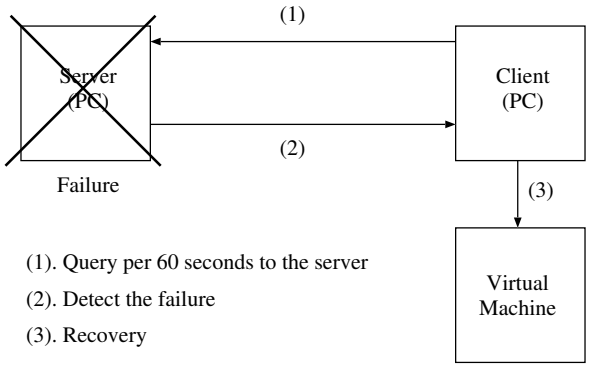

Fig. 4. Recovery in the proposed method.

of applications by the proposed methods. Then, the backup is sent to another client (a spare node) regularly. Note that the files of the backup are described the execution environment of the hardware (platform) used at that point. Thus, if the CA on a different hardware has recovered from the backup, the behavior of its OS becomes unstable due to disagreement of architectures or conflict of device drivers. In the worst case, the OS cannot work. On the other hand, to solve the problem by using virtual machine based on virtualization techniques, there are advantages as follows:

- A backup is easily facilitated, because the whole executable environment can gather in several files.
- Failure can hardly occur in the environment, because a virtual hardware (virtual machine) based on virtualization absorbs the difference of the environments.

Figure 1 shows the overview of the proposed methods. Concerning about the advantages mentioned above, the CA is built not on the environment in the hardware directly, but on the guest OS of the virtual machine [5]–[6]. Thus, fault tolerance makes it possible to obtain in three steps as follows:

Step.1 From the main host which is working normally, the backup of the environment in the virtual machine is sent to a spare host as shown in Fig. 1 (1).

Step.2 The existence of the main host is overseen by watchdog programs as shown in Fig. 1 (2).

Step.3 If the main host has been down, the backup is invoked by the watchdog program of the client as shown in Fig. 1 (3). As a result, a virtual machine in the spare host boots up.

Note that VMware [5] is used for an example to make a virtual machine in this paper. The spare host may have the same performance characteristics as the main host. The proposed approach uses the virtualization in every server that is running a Grid Service. This may be a great source of performance degradation since it introduces a new abstraction layer. However, because of the recent advanced CPUs such as dual-core or multi-core processors, the performance degradation is considered to make it possible to control to some extent. The proposed approach is superior to other approaches in the low-cost point of view. For example, since the existing backup-recovery techniques are possible to adopt the proposed methods, the dependable gird system is easily implemented. However, the cost will take to implement such a database method in the grid system by including the development of the database program explicitly or implicitly.

B. Backup Procedure

Fig. 2 shows the backup in the proposed method. The procedures of the backup in the virtual machine are as follows: Specifically, a backup, which contains some files in a host OS, is gained and it can build a virtual machine based on the VMware.

Table I lists the files and its volumes for the backup procedure. The first column of file in Table I presents a NVRAM file, a VMSD file, a virtual memory disk file, a virtual memory disk file, and a virtual machine configuration file, respectively. Note that the files are copied in a spare node by a software tool such as BunBackup (version 2.61) through a network. Since the files were assigned for backup beforehand and they were set up to copy the files from a server to a client per two hours by BunBackup, the backup is performed automatically.

C. Failure/Fault Detection Procedure

The procedures of detecting a failure or fault in the CA are described in the following. The detection of the failure/fault is performed to oversee by watchdog programs whether the CA is down or not through a network. If the CA is not identified by the programs, we regard it as a failure or fault has occurred. Fig. 3 shows an outline of detecting a failure or fault. In this paper, to detect a failure or fault, a client-server program is used. The program for the detection is implemented in a client and server, respectively. Note that the client is trying to send a query per 60 seconds to the server by a socket communication based on Winsock. After the server received the query, it is then to send an ACK to the client for its alive.

TABLE I
FILES AND THEIR VOLUMES.

File	Its volume
Other Linux 2.6.x kernel.nvram	9KB
Other Linux 2.6.x kernel.vmsd	0KB
Other Linux 2.6.x kernel-flat.vmdk	16,777,216KB
Other Linux 2.6.x kernel.vmdk	1KB
Other Linux 2.6.x kernel.vmx	1KB

D. Recovery Procedure

When a failure or fault is detected, the CA is recovered by starting a virtual machine (a spare host) from the configuration files of the backup. In the case that a client cannot connect to a server which plays a role of the CA as described in the previous section, the client regards the server as to be faulty. Thus, a fault has occurred in the CA. The client also invokes a virtual machine from the configuration files of the backup in itself. The following source codes are a part of the program.

```
/*trying to connect to a server;
  if it is error, then a file is opened */
if (connect(sock, (SOCKADDR *)&sockaddr,
      sizeof(sockaddr)) != SOCKET_ERROR) {
      recv(sock, buf, MAX_PATH, 0);

}
```

The program of the client is trying to be a socket communication for that of the server. If the program fails in the connection, the configuration files of the backup will be opened. Concerning the server, an IP address, a port number, and a file name are described in a `ini_file`. The `ini_file` is located at the same directory as the program.

For example, an IP address is `192.168.1.5`, a port number `1024`, and a file `test.txt` to be run are written in the `ini_file`. First, the program is trying to connect the port number `1024` of the IP address `192.168.1.5`. Then, the file `test.txt` is executed if the socket communication has failed.

In practice, to recover a virtual machine using the program automatically, a configuration file of the virtual machine (e.g., `Other Linux 2.6.x kernel.vmx`) is beforehand given over to the program of the virtual machine (e.g., `Vmwareplayer.exe`), so that a virtual machine can run automatically by this process. Fig. 4 shows an example of the recovery of the process. On the other hand, if the virtual machine fails to start itself, it needs to try it again.

Assumed that a failure/fault occurs in a host, the system survives up to $n-2$ failures, where n is the number of hosts. This is because at least two hosts must need for the detection/recovery of the failure/fault in the system.

IV. EVALUATIONS

In this section, the environment for computer simulations, the way for the simulations, and the results are described.

A. Simulation Method

To realize the proposed dependable grid system, a grid environment was built by using the Globus Toolkit 4 [3].

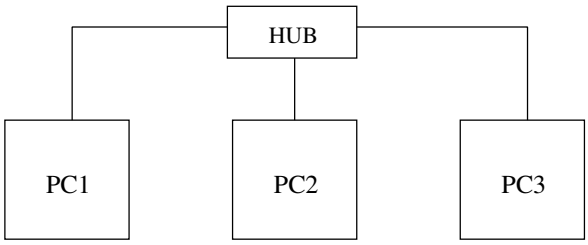

Fig. 5. Simulation environment.

Table II shows each computer organization for building the grid environment. Note that two entries with parenthesizes in Table II are ones after recovering the CA.

Fig. 5 shows the simulation environment. In practice, three computers, denoted by PC1, PC2, and PC3, were used for the simulations as shown in Fig. 5. The grid environment was composed of two computers (PC1 and PC3). Assumed that a fault occurs in the PC1. The PC2 was used for a backup and recovery in simulations. Note that the PC2 can process ordinary jobs. Also note that the Globus Toolkit 4 was installed on the Fedora Core 4. The VMware Workstation 5.5.3 was used for virtualization as software. The user accounts for the use of the Globus Toolkit 4 and their roles are as follows:

- **root** as an administrator for hosts;
- **globus** as normal users for executing and installing the Globus Toolkit 4; and
- **gtuser** as any users for using services in the Globus Toolkit 4.

For the proposed system in Section III and the grid environment in Subsection IV-A, simulations were performed to verify the fault tolerance of the system in the following. Note that a virtual machine being built on the PC1 has its function of the CA and that of executing processes from other nodes. Also note that the PC3 is an ordinary node without the CA.

First, the backup was executed from the PC1 to the PC2 by the procedures as described in Subsection III-B. Then, the watchdog program as described in Subsection III-C and the recovery program as described in Subsection III-D were executed in the PC1 as a server and in the PC3 as a client. Finally, the PC1 made it down intentionally. For the PC being recovered automatically, two operations (i.e., A and B) were tried to execute as follows:

Operation A: Whether the recovered CA can add a node normally or not.

Operation B: Whether the recovered CA can request a process using a certification mutually or not.

Concerning to the operation A, each kind of authentication was executed. Also to the operation B, a job was thrown using a definition file which used for checking the system in setting up the GRAM of the Gloubus Toolkit 4.

B. Simulation Results

In this subsection, simulation results are shown as follows: The PC1 (CA) made it down intentionally, so that we confirmed that a new CA can recover from the backup on the

TABLE II
COMPUTER ORGANIZATIONS FOR BUILDING A GRID ENVIRONMENT.

Computer	CPU	Memory	HDD	Host OS	Guest OS	Host name	Line speed
PC1	Athlon64 3500+	1024MB	160GB	Windows XP	Fedora Core 4	grid1.elfox.org	100Mbps
PC2	Pentium4 2.6GHz	512MB	80GB	Windows XP	(Fedora Core 4)	(grid1.elfox.org)	100Mbps
PC3	Pentium4 2.4GHz	512MB	80GB	Fedora Core 4	none	grid2.elfox.org	100Mbps

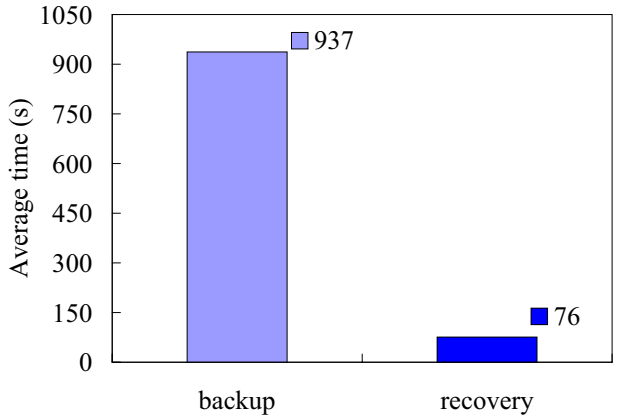

Fig. 6. Simulation Results.

PC2 by the watchdog program in two operations presented in the previous subsection as follows:

In the **Operation A**, whether the recovered CA can add a node normally or not. The result was that the CA can do the new node. In the **Operation B**, whether the recovered CA can request a process using a certification mutually or not. The result was that the CA can do the process. As a result, we confirmed that the recovered CA can publish the certificate and it can add the new node (PC3). In addition, the PC3 can process using the certificate from the PC2.

The time taken the backup was measured the mean of 10 times in each backup repeatedly. As a result, the time of the backup was 937 seconds as shown in Fig. 6. Note that the backup time is from the time which the files are started transmitting to the time which all of the files are finished in transmitting by a back up tool such as software.

The time taken the recovery was measured the mean of 10 times in each recovery repeatedly. As a result, the time of recovery was 76 seconds as shown in Fig. 6. Note that the recovery time is from that the CA is booting to that it is to accept a process on the PC2. Also note that the starting time of the recovery and the finished time of that, respectively, are measured from a watchdog program by a shellscript which put on a virtual machine.

V. DISCUSSION

From the simulation results, since it is possible to add a new node and to process mutual authentication, we can achieve to build the dependable grid system. In the following, the backup time, the recovery time, advantage in auto-recovery, a watchdog program of virtual machine, the recovery from a half-finished backup, and the application level fault-tolerance are discussed.

- **Backup Time**: Since the backup is executed through a network which is connected by the 100Mbps LAN, the speed of transmitting data could be a bottleneck. Thus, the backup time is considered to reduce by a high speed LAN.
- **Recovery Time**: Since the watchdog programs are confirmed the alive of nodes per 60 seconds, the time is passed at most 60 seconds if the node has been down as soon as after the confirmation. Consequently, since the average time is 90 seconds until a virtual machine has invoked and its process has finished, the recovery time becomes around from 90 seconds to 210 seconds.
- **Advantage of Auto-recovery**: In this paper, the CA is recovered automatically by a watchdog program. However, in the case that the CA is not needed for other nodes immediately, the CA may recover by a manual operation. On the other hand, in the case that the CA is needed for one node, which executes its process and its data are stored on the CA, the CA should be recovered by automatically.
- **Surveillance of Virtual Machine**: Since the watchdog programs are running on the OS in a server and client, the alive of the virtual machine is not watched itself. This reason is the use of a socket communication by the Winsock program so that another platform may be not worked. Thus, the program which can run multiplatform should be developed.
- **Recovery from the half-finished Backup**: In the case that the CA is with its backup half-finished done and is down, the recovery process is not worked correctly, because the files of the backup are always stored as the files are overwritten. Thus, the files of the backup are not stored correctly. To solve this problem, the backup files are copied periodically and old files are stored until new files are stored completely. So that a watchdog program can recover the virtual machine safely.
- **Application level fault-tolerance**: Since the proposed approach does not deal with the data on processing are lost if a failure has occurred in the running application, the lost data cannot be recovered. Thus, to avoid this, fault tolerance at the application level will be needed to implement by using checkpointing, replication, and database methods.

VI. CONCLUSIONS

In this paper, we proposed an off-the-shelf scheme for dependable grid systems using virtualization. The proposed approach can apply it to heterogeneous environments by using virtualization. Since backup and recovery use their existing techniques concerning about commodity off the shell, fault tolerance can be easily achieved. Simulation results show that the

system using virtualization can tolerate a failure/fault such as hardware and software. Future research issue on the proposed methods remains to be explored: this includes enhancing more availability and dependability in using checkpointing methods or other related methods.

ACKNOWLEDGMENT

The author would like to thank Mr. Satoru Yamamoto, who was a member of the laboratory of the author, for his support on this work.

REFERENCES

[1] I. Foster and C. Kesselman, ed., *The Grid 2: Blueprint for a new computing infrastructure*, Morgan-Kaufman, 2003.
[2] Ian J. Taylor, *From P2P to web services and grids: Peers in a client/server world*, Springer, 2005.
[3] The Globus Alliance, http://www.globus.org/
[4] Steven J. Vaughan-Nichols, "New approach to virtualization is a lightweight," *IEEE Computer*, vol.39, no.11, pp.12–14, Nov. 2006.
[5] J.E. Smith and R. Nair, *Virtual machines Versatile Platforms for Systems and Processes*, Morgan Kaufmann, 2005.
[6] M. Rosenblum and T. Garfinkel, "Virtual machine monitors: Current technology and future trends," *IEEE Computer*, vol.38, no.5, pp.39–47, May 2005.
[7] S. Hwang and C. Kesselman, "A flexible framework for fault tolerance in the grid," *J. Grid Computing*, vol.1, no.3, pp.251–272, 2003.
[8] X. Zhang, D. Zagorodnov, M. Hiltunen, K. Marzullo, and R.D. Schlichting, "Fault-tolerant grid services using primary-back-up: feasibility and performance," *Proc. 2004 IEEE Int'l. Conf. Cluster Computing (Cluster 2004)*, pp.105–114, 2004.
[9] P. Townend and J. Xu, "Dependability in Grids," *IEEE Distributed Systems Online*, vol.6, no.12, pp.1–7, Dec. 2005.
[10] Y. Horita, K. Taura, and T. Chikayama, "A scalable and efficient self-organizing failure detector for grid applications," *Proc. 6th IEEE/ACM Int'l. Workshop Grid Computing (GRID'06)*, pp.202–210, 2006.
[11] H. Jin, X. Shi, W. Qiang, and D. Zou, "DRIC: Dependable grid computing framework," *IEICE Trans. Inf.& Syst.*, vol.E89-D, no.2, Feb. 2006.
[12] Y.S. Dai, Y. Pan, and X. Zou, "A hierarchical modeling and analysis for grid service reliability," *IEEE Trans. Computer*, vol.56, no.5, pp.681–691, 2007.
[13] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh, "Terra: A virtual machine-based platform for trusted computing," *Proc. 19th ACM Symp. Operating Systems Principles (SOSP'03)*, vol.37, Issue 5, pp.143–206, 2003.
[14] I. Krsul, A. Ganguly, and J. Zhang, "VMPlants: Providing and managing virtual machine execution environments for grid computing," *Proc. 2004 ACM/IEEE Conf. Supercomputing (SC'04)*, p.7, 2004.
[15] M. Zhao, J. Zhang, and R. Figueiredo, "Distributed file systems support for virtual machines in grid computing," *Proc. 13th IEEE Int'l. Symp. High Performance Distributed Computing (HPDC'04)*, pp.202–211, 2004.
[16] E. Kim, H.S. Kim, H.Y. Yeom, and J. Lee, "GiSK: Making secure, reliable and scalable VO repository virtualizing generic disks in the Grid," *Proc. 8th Int'l. Conf. High-Performance Computing in Asia-Pacific Region (HPCASIA'05)*, pp.370–377, 2005.
[17] P. Suranyi, H. Abe, T. Hirotsu, Y. Shinjo, and K. Kato, "General virtual hosting via lightweight user-level virtualization," *Proc. 2005 Symp. Applications and the Internet (SAINT'05)*, pp.229–236, 2005.
[18] V. Buge, Y. Kemp, M. Kunze, and G. Quast, "Application of virtualization techniques at a university grid center," *Proc. 2nd IEEE Int'l. Conf. e-Science and Grid Computing (e-Science'06)*, p.155, 2006.
[19] B. Sundaram and B.M. Chapman, "A grid authentication system with revocation guarantees," Technical Report Number UH-CS-05-24, Computer Science Department, University of Houston, Jan 25, 2006.
[20] K. Kagawa, K. Yamada, T. Kamiya, and M. Nagata, "Fault tolerant grid migration using network storage," *Proc. Int'l. Conf. Parallel and Distributed Processing Techniques and Applications (PDPTA'06)*, vol.1, pp.241–245, 2006.
[21] S. Tadepalli, C. Ribbens, and S. Varadarajan, "GEMS: A job management system for fault tolerant grid computing," *Proc. Int'l. High-Performance Computing Symp.*, pp.59–64, 2004.