PROCEEDINGS OF SCIENCE

Secure Peer to Peer Message Passing using A-JUMP

Mehnaz Hafeez
[1] *Shaheed Zulfikar Ali Bhutto Institute of Science and Technology (SZABIST)*
Street # 09, Plot # 67, Sector H-8/4, Islamabad, Pakistan
E-mail: Mehnaz.Hafeez@cern.ch

Sajjad Asghar*[1]
[2]*National Centre for Physics (NCP)*
Quaid-i-Azam University Campus, Islamabad, Pakistan
E-mail: sajjad@ncp.edu.pk

Usman A. Malik[2]
E-mail: usman@ncp.edu.pk

Adeel-ur-Rehman[2]
E-mail: adeel@ncp.edu.pk

Naveed Riaz[1]
E-mail: n.r.ansari@szabist-isb.edu.pk

MPI is a de facto standard for message passing for high performance parallel, as well as, for distributed computing environment. The static and homogenous model of MPI is not compatible with the dynamic and heterogeneous Grid environment. There are not many implementations which offer message passing over Internet and Grids. P2P-MPI and A-JUMP are MPI implementations, which provide both point-to-point and collective data operations over Internet using Java. However, none of these MPI implementations are categorized as a secure message passing implementation. In these MPI implementations security is overlooked to avoid performance degradation. Security becomes a fundamental concern when considering message passing over Internet, and should be accordingly addressed. Moreover, access over Internet requires authorization/authentication of resources and users along with encryption of data. In this paper, we have proposed secure message passing over Internet and Grids using A-JUMP. This paper also presents performance analysis and comparison of the proposed model with P2P-MPI and A-JUMP.

The International Symposium on Grids and Clouds and the Open Grid Forum
Academia Sinica, Taipei, Taiwan
March 19 - 25, 2011

*[1] Sajjad Asghar

PoS(ISGC 2011 & OGF 31)024

1. Introduction

Programmers and researchers need an interoperable, synchronous and reliable working environment to develop high performance applications. It is difficult to run MPI enabled applications on Grids where availability of different type of resources keeps changing. Moreover, available MPI implementations are least concerned about the security of resources and user processes because it increases communication overheads for clusters. Therefore, it is still a challenge to provide a facility of secure message passing without losing the performance. For Grids, security is an essential component for message passing over Internet. Security components are an important part of the implementation of all Grid and peer-to-peer computing middleware to provide secure communication. Currently A-JUMP for interconnected clusters [16] and P2P-MPI [12] are available message passing frameworks that offer peer-to-peer message passing over Grids and Internet. However, both A-JUMP and P2P-MPI have ignored message security. In this paper we have extended A-JUMP model over Internet [16] for secure message passing using the available security techniques. A-JUMP for interconnected clusters is preferred since it effectively utilizes wide-area inter-cluster networks [16]. A-JUMP [15] uses a novel communication model for message passing called High Performance Computing (HPC) bus that uses JMS for communication and message passing in an asynchronous manner.

The paper is organized as follows: section 2 covers the literature overview of different security models of MPI and Grid, details of secure A-JUMP are presented in section 3, performance analysis is presented in section 4 and the concluding remarks and future directions are covered in section 5.

2. Literature Review

Since the security models for MPI are not many and proposed model is closer to Grid like, therefore, in research review GRID security models are also included. This section covers the literature review of existing MPI and Grid security models.

The security of MPI is critical. MPI applications over Internet require an extensive range of security policies because of the open nature of Internet. A safe message passing of processes is a defined feature of MPI [17] but both MPI and MPI-2 do not address the security policies on the wide spectrum for distributed HPC applications.

There is not enough work carried out for secured MPI, though, there are many implementations of MPI and MPI-2 available [18]. The general approach to improve security of MPI applications may allow application programmers themselves to include message confidentiality that reduces portability and flexibility.

On the other hand, the MPI interface can be extended for APIs, like MPISec I/O [13]. MPISec I/O supports message passing with data confidentiality and programmers can manually set the required encryption rules. However, if a programmer does not set up encryption and decryption rules carefully in the code, it is possible that some data is stored without encryption or read without decryption. Furthermore, it is not compatible with non-secure MPI libraries that make preceding programs non-functional unless they are updated according to the extended libraries.

Security model for OGSA is based on GT-3 and web-services protocols [1]. The proposed security model shows improvements over previous GT-2 model. It is based on least privilege model. This model is not generalized enough and its implementation is Globus specific. On the other hand, the ASCI project ports Globus system from GSI to Kerberos [2]. GSS-API layer modifications provide interoperability between GSI & Kerberos. The solution lacks adaptability and reusability.

Both [5] and [7] have attempted to map role based access techniques on Grid security. [5] focuses on information sharing and security risks. It is a theoretical model that lacks implementation and validation. dRBAC provides credential discovery, validation, delegation and management in distributed environment [7]. However, it has no provision to limit the transitive trust. PERMIS [8] is another model that is based on RBAC like [5] and [7]. It focuses on authorization only. Its practical implementation is available with generic APIs.

Like [1], Grid security model based on OGSA [6] has mapped web services security models to Grid security. The proposed security model broadens the existing security techniques for web services. The implementation and validation of the model is not presented. [10] is based on Globus middleware and provides method level access with credential delegation. [9] is also specific to Globus middleware. The UCON model has been extended to provide usage control. It provides a generic architecture, active policy decision and dynamic authorization policy.

[11] is based on Legion. It has focused on delegation of credentials and authorization. TRMS [3] is a trust model for Grids and trust-aware resource management system. It has reduced security overheads, thus, improving Grid performance. Unlike other security models, it uses a heuristic based algorithm. It is observed that all Grid security models assure infallible authorization and authentication mechanisms; however, they do not address the secure message passing.

3. Secure A-JUMP

A-JUMP model for interconnected clusters [16] requires addressing the security related to message passing in a comprehensive manner. Secure message communication requires confidentiality, integrity, authentication and non-repudiation. Through encryption, the confidentiality is achieved. The symmetric algorithm called AES is used to encrypt the data and RSA is used for message digest. Authorization and authentication is achieved using X509 certificate. To achieve the integrity and non-repudiation PK-Grid-CA [14] certificate is used. All these features need to be integrated with A-JUMP. HPC bus of A-JUMP is based on ActiveMQ [4]. Since ActiveMQ does not provide message level encryption and decryption, therefore, Secure A-JUMP (SA-JUMP) layer is introduced in HPC bus over Active-MQ.

SA-AJUMP encompasses security needs of all participating entities. This includes users, applications, resources and resource owners. Security Format is based on X509 certificates and Secure Sockets Layer (SSL). Public Key Infrastructure (PKI) is preferred essentially for the reason that A-JUMP may run on Grid. The Message Security module of A-JUMP communicates with the SA-JUMP layer. It focuses on the following fundamental goals; protection of credentials, interoperability with local site security infrastructures and secure message passing along with user and host authentication and authorization.

3.1 SA-JUMP Architecture

Layered Architecture of SA-JUMP for interconnected clusters is presented in Figure 1. SA-JUMP Layer consists of four modules. These are as follows:

- Key Store Manager (KSM)
- Message Encryption Manager (MEM)
- Secure Resource Allocation (SRA)
- Secure Process Resource Allocation (SPRA)

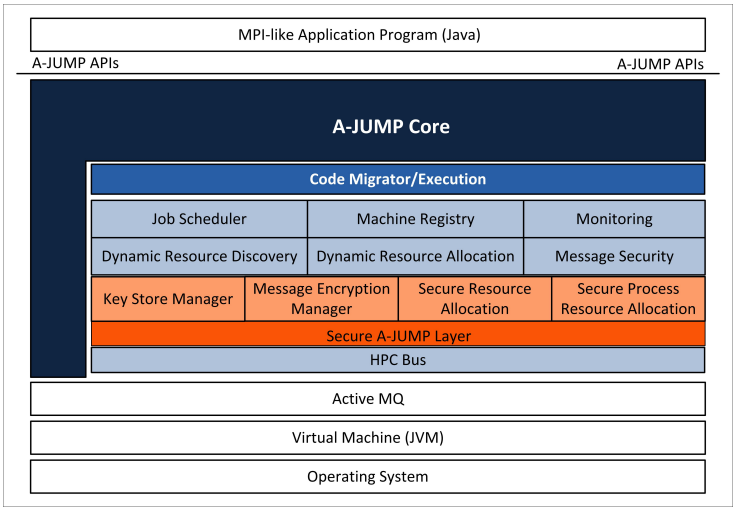

Figure 1: Layered Architecture of SA-JUMP

3.1.1 Key Store Manager (KSM)

KSM is responsible for central key management. It is responsible for key validation. KSM maintains key store and its passphrase. Key store contains hosts and users keys. Both, the host and user keys are issued by trusted Certificate Authority. KSM also distributes and redistributes the key store if any key is added, revoked, expired, issued or reissued. In current implementation all users use the host key for authorization and authentication of resources.

3.1.2 Message Encryption Manager (MEM)

MEM does encryption and decryption using agreed keys. It is also responsible for session key exchange required for data encryption and decryption. It takes information directly from KSM. MEM uses mixed mode where user encrypts the data using his private key and then encrypts one more time by using host public key.

3.1.3 Secure Resource Allocation (SRA)

SRA ensures authentication for new resources to become a part of A-JUMP framework. Moreover, it ensures whether entity is authorized to use the resources or not according to the predefined security policies.

3.1.4 Secure Process Resource Allocation (SPRA)

SPRA uses APIs for secure send and secure receive. It utilizes session key for secure message communication. It creates and maintains a secure virtual cluster consisting of resources

required for secure communication over Internet. Detailed component level diagram is presented in Figure 2.

3.2 SA-JUMP Workflow

The main components of SA-JUMP layer are KSM, MEM, SRA, and SPRA. Each machine in cluster must run Code Migrator/Execution (CME) Adaptor to receive the incoming program files. A machine may serve as a CME Adaptor, as well as, a client.

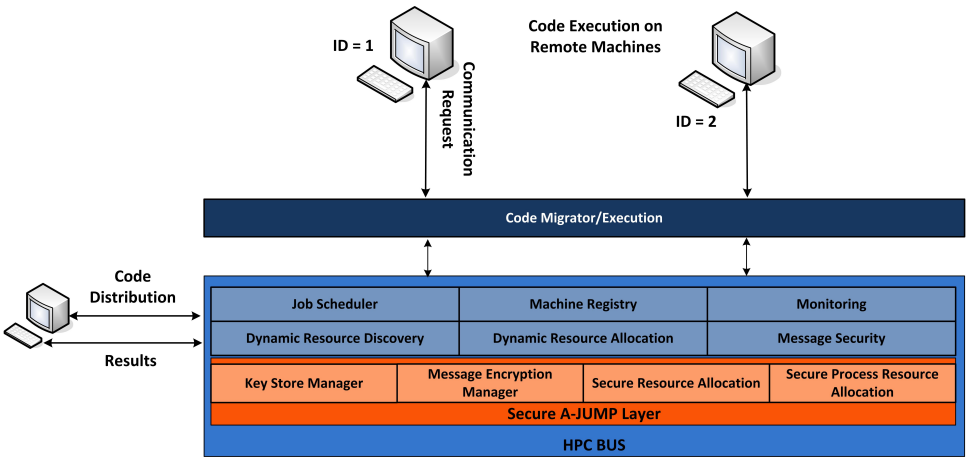

Figure 2: Component Level Diagram of SA-JUMP

Monitoring keeps track of the status of resources that are part of A-JUMP clusters. The status can either be free or busy. MR is responsible to send information to Monitoring in case a new resource is added, updated, and deleted. Also when the status changes from free to busy or vice versa MR sends the updates to Monitoring.

To enable secure message passing SRA setups a pool of available resources for secure communication. SPRA gets this information from DRD and DRA of HPC bus. Key exchange between the resources is done by MEM using RSA at the same time. This is how a virtual cluster is created for secure message passing. SPRA includes ssend() and sreceive() APIs for sending and receiving secure messages. The calls ssend and sreceive exchange information with MEM to encrypt and decrypt the data.

When a new resource is added to A-JUMP cluster it is assigned a unique rank by the MR. SRA requests for the host certificate, to authenticate resources, so they may become a part of A-JUMP clusters. The key validation is done through KSM, which is responsible for key management. Once the new resource is registered, this information is propagated to the rest of the central components, which include DRD, DRA and SRA through Monitoring.

Client needs to request for number of processes required to run the job. Before the job is submitted to the cluster/clusters, the user sends this request to the DRD/DRA for the given number of free processes. DRD/DRA seeks the information of available free resources from Monitoring. It searches the indexed free processes for sub-clustering.

Once client receives the information about the required free resources, it submits the job to the specific machines identified by the DRD/DRA. These machines may reside on different clusters or on a single cluster. When CME Adaptors start execution of the submitted job, they inform MR about the change in the status from free to busy. At the same time MR also sends

the updates in the change of status to the Monitoring. As soon as the submitted job finishes, a request of change in status from busy to free is sent back to MR. CME Adaptors are also responsible to send results back to the client. Figure 3 illustrates the relationship and the information shared between the different components. It also describes the step by step flow of program running in parallel on different clusters.

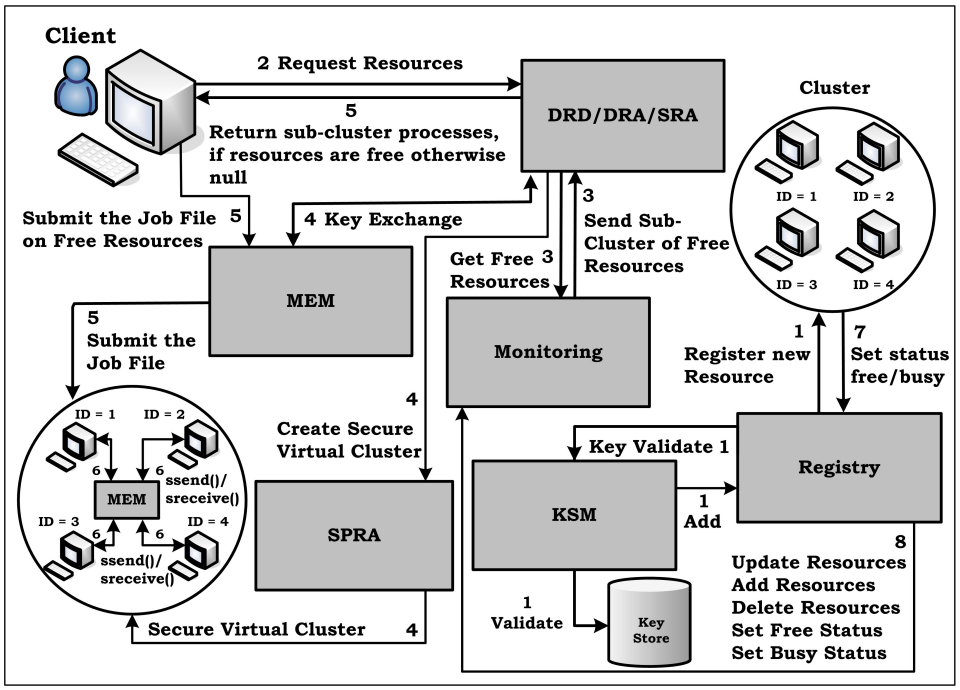

Figure 3: Workflow of Job Submission using SA-JUMP

4. Performance Analysis

The current enhancement presented in this paper affects the communication mechanism of A-JUMP, therefore, point-to-point communication performance measurement results using ping-pong latency test and network throughput measurement are included.

The tests are repeated 1000 times for various message sizes between two different machines in two different clusters running on different Internet domains. Machines are randomly selected from different clusters. The encryption algorithms RSA and AES are used. RSA is used for key exchange, user authentication and authorization of resources. AES is used for data encryption and decryption. The results of SA-JUMP for secure message passing are compared with P2P-MPI and A-JUMP for interconnected clusters.

4.1 Test Environment

Results were collected on an environment that simulated globally interconnected clusters. The hardware resources included in conducting the tests were; 9 single core machines with 3.2GHz single core CPUs, 1GB RAM and a Gigabit LAN interface; divided into two clusters. The clusters were connected with 1 Gbps bandwidth. The machines had Windows Server 2003 (with SP2) and Windows XP (with SP3) running on them. The TCP window size was default on

all the machines in clusters. There was no optimization done at the level of OS, as well as, at the level of hardware.

4.2 Communication Performance Measurement

Ping-pong communication test presents latency results on unidirectional sending and receiving of secure messages. Figure 4 shows the comparison of ping-pong latency between P2P-MPI, A-JUMP and SA-JUMP. The results demonstrate that SA-JUMP has introduced some encryption and decryption overheads for message passing since it shows higher latencies as compared to A-JUMP for interconnected clusters.

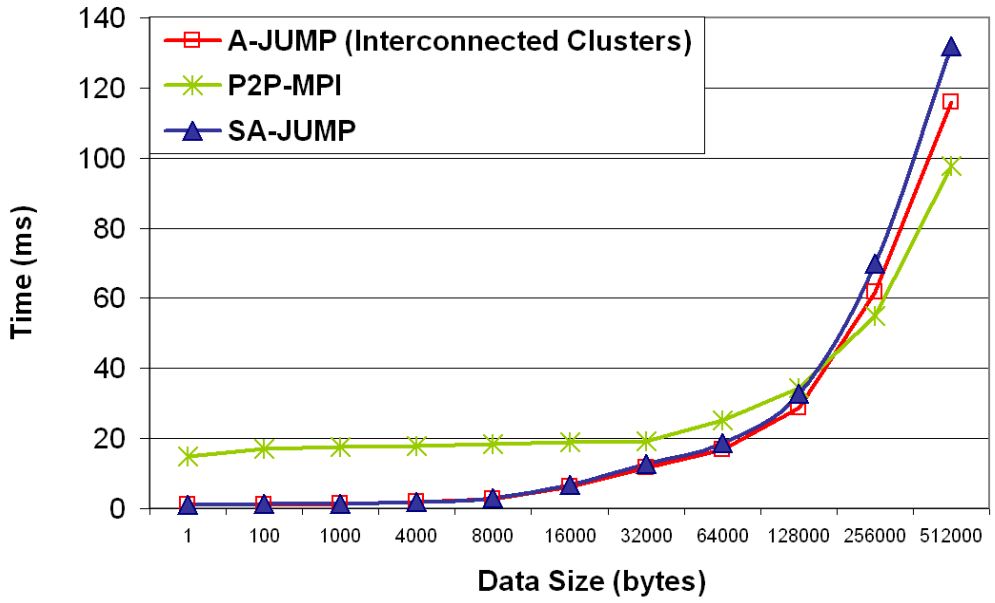

Figure 4: Ping Pong Latency Measurements of SA-JUMP

SA-JUMP layer reduces the bandwidth consumption significantly as compared to A-JUMP itself. The network throughput results in Figure 5 show that for message size of $8x10^3$ bytes, SA-JUMP has consumed network bandwidth that is 15% less than A-JUMP for interconnected clusters. This consumption further decreases up to 20% as data size increases up to $256x10^3$ bytes. Thereafter, it remains constant.

Both A-JUMP for interconnected clusters and SA-JUMP have performed better than P2P-MPI up to the message size of $128x10^3$ bytes. For message sizes larger than $128x10^3$ bytes, P2P-MPI performs better because the current implementation has inherited shortcoming of A-JUMP, which is not optimized to handle the communication for the large message sizes [15]. This limitation on A-JUMP is imposed by the ActiveMQ, which cannot send large object messages efficiently.

5. Conclusion

SA-JUMP is a feasible secure message passing solution when security and confidentiality of data is required. The secure message passing in A-JUMP for interconnected clusters is achieved through PKI infrastructure and X509 certificate. Secure A-JUMP (SA-JUMP) Layer

provides authorization, authentication, sending and receiving of secure messages over the network.

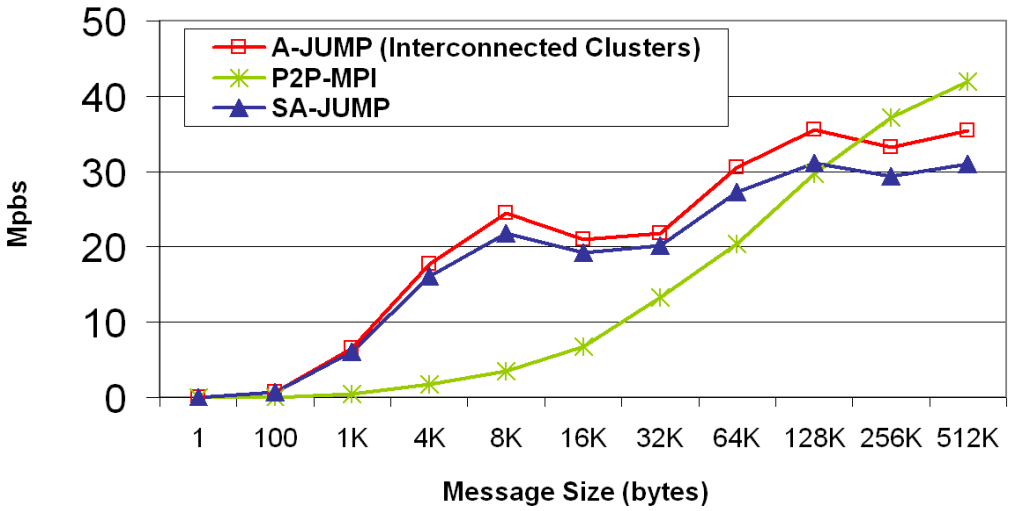

Figure 5: Network Throughput Measurements of SA-JUMP

Performance overheads are observed because the security layer is introduced to provide encryption and decryption. The performance results show that SA-JUMP has introduced considerable overheads in comparison with A-JUMP for interconnected clusters. Furthermore, SA-JUMP has performed better than P2P-MPI for smaller message sizes and become comparable for the message size of 128×10^3 bytes. The results are promising and can be improved for larger message sizes.

References

[1] V. Welch, F. Siebenlist, I. Foster et al., *Security for Grid Services*, in proceedings of *12th IEEE International Symposium on High Performance Distributed Computing (HPDC-12)*, Seattle, Washington, June 2003.

[2] Patrick C. Moore, Wilbur R. Johnson, Richard J. Detry, *Adapting Globus and Kerberos for a Secure ASCI Grid*, in proceedings of *ACM/IEEE Super Computing Conference*, 2001, pp: 54.

[3] F. Azzedin, M. Maheswaran, *Towards Trust-aware Resource Management in Grid Computing Systems*, in proceedings of *Cluster Computing and the Grid 2nd IEEE/ACM International Symposium CCGRID2002*, 2002, pp: 452 – 457.

[4] ActiveMQ website. [Online]. http://activemq.apache.org.

[5] C. E. Phillips, T. C. Ting, S. A. Demurjian, *Mobile and Cooperative Systems: Information Sharing and Security in Dynamic Coalitions*,in proceedings of *7th ACM Symposium on Access Control Models and Technologies*, June 2002.

[6] V. Mukhin, *The Security Mechanisms for Grid Computers*, in proceedings of *4th IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS 2007)*, Sept. 2007, pp: 584 – 589.

[7] E. Freudenthal, T. Pesin, L. Port, E. Keenan, and V. Karamcheti, *dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments*, in proceedings of *22nd International Conference on Distributed Computing Systems (ICDCS)*, IEEE Computer Society Press, 2002, pp: 411-420.

[8] D. W. Chadwick A. and Otenko, *The PERMIS X.509 Role Based Privilege Management Infrastructure*, in proceedings of *7th ACM Symposium on Access Control Models and Technologies*, 2002.

[9] F. Martinelli, P. Mori, *A Model for Usage Control in Grid Systems*, in proceedings of *Grid-STP 2007, International Workshop on Security, Trust and Privacy in Grid Systems*, IEEE, 2007.

[10] H. Jung, H. Han, H. Jung, H. Y. Yeom, *Flexible Authentication and Authorization Architecture for Grid Computing*, in proceedings of *International Conference on Parallel Processing (ICPP 2005),* June 2005, pp: 61 – 77.

[11] G. Stoker, B. White. E. Stackpole, T.J. Highley, and M. Humphrey, *Toward Realizable Restricted Delegation in Computational Grids*, in proceedings of *International Conference on High Performance Computing and Networking Europe (HPCN Europe 2001)*, Amsterdam, NL, June 2001.

[12] S. Genaud and C. Rattanapoka, *P2P-MPI: A Peer-to-Peer Framework for Robust Execution of Message Passing Parallel Programs* in *Journal of Grid Computing*, 2007, pp: 27 – 42.

[13] R. Prabhakar, C. Patrick, and M. Kandemir, *Mpisec I/O: Providing data confidentiality in MPI-I/O*, in *Cluster Computing and the Grid, IEEE International Symposium*, 2009, pp: 388 – 395.

[14] PK-Grid-CA website. [Online]. http://www.ncp.edu.pk/pk-Grid-ca.

[15] S. Asghar, M. Hafeez, U. A. Malik, A. Rehman and N. Riaz, *A-JUMP Architecture for Java Universal Message Passing*, in proceedings of *8th International Conference on Frontiers of Information Technology (FIT 2010)*, Islamabad, Pakistan, 2010, doi:[10.1145/1943628.1943662].

[16] M. Hafeez, S. Asghar, U. A. Malik, A. Rehman, and N. Riaz, *Message Passing Framework for Globally Interconnected Clusters*, in *International Conference on Computing in High Energy and Nuclear Physics (CHEP 2010)*, Taipei, Taiwan, 2010, unpublished.

[17] MPI: A Message Passing Interface Standard. Message Passing Interface Forum website. [Online]. http://www.mpi-forum.org/docs/mpi-11-html/mpi-report.html.

[18] M. Hafeez, S. Asghar, U. A. Malik, A. Rehman, and N. Riaz, *Survey of MPI Implementations*, in *International Conference on Digital Information and Communication Technology and its Applications (DICTAP 2011)*, Dijon, France, CCIS/LNCS, vol. 167, 2011, pp: 1011–1025.