International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:8, No:10, 2014

Enhancing Security in Resource Sharing Using Key Holding Mechanism

M. Victor Jose, V. Seenivasagam

Abstract**—**This paper describes a logical method to enhance security on the grid computing to restrict the misuse of the grid resources. This method is an economic and efficient one to avoid the usage of the special devices. The security issues, techniques and solutions needed to provide a secure grid computing environment are described. A well defined process for security management among the resource accesses and key holding algorithm is also proposed. In this method, the identity management, access control and authorization and authentication are effectively handled.

Keywords**—**Grid security, Irregular binary series, Key holding mechanism, Resource identity, Secure resource access.

I. Introduction

IN general, a grid can provide a consistent way to balance the loads on a wider federation of resources applying to CPU, storage, and other types of resources that may be available on a grid [1]. Authentication and Access Control is a major issue in grid security for resource usages. In the grid environment, if a node (GNi) has a secure file S, all the other nodes in the grid can access the secure file and the attack is highly possible [1]. Also if the resource S has to limit for a particular user, it will be a tedious job to hold the key of the limited users. Ontology and key servers are introduced to do the job. The proposed architecture for a secure resource handling is a logical method to handle the secure resources to avoid the attacks. To handle the resources, a new identity management system is introduced. A secure file access control protocol through the middleware of grid environment is used. The Grid service providers need to make sure that the resource cannot be accidentally accessed by another user or a hacker. The right level of security is necessary to use a resource provider's offerings. A standard secure grid architecture is introduced to make the resources to be private for the users. In a grid, the member machines are configured to execute programs rather than just move data which make an unsecured grid potentially fertile ground for viruses and Trojan horse programs. It is important to understand exactly which components of the grid must be rigorously secured for the aforesaid reason. Furthermore, it is necessary to understand the issues involved in authenticating users and providing proper authorization for specific operations. Managing a large group of computing is a primary task in the grid environment as well as the secure resource handling [2]. In the proposed architecture, all the above problems are analyzed and security mechanism is implemented using Gridsim Toolkit 5.2 [3].

M.Victor Jose is with the Computer Science and Engineering Department, Noorul Islam Centre for Higher Education, Kumaracoil, Tamil Nadu, India (Mob: +919487112584; Fax: +914651257266; e-mail: victorjose@yahoo.com).

V. Seenivasagam is with the Computer Science and Engineering Department, National Engineering College, Kovilpatti, Tamil Nadu, India (Mob: +919442622502; Fax: +914632232749; e-mail: yespee1094@yahoo.com).

II. Materials and Methods

Secure resource sharing has been enhanced using different types of key authentication protocols such as Ontology based, Database based, and Kerberos based [4]-[7], [13]. Reference [6] proposed encryption algorithm is to make an authenticator. Here, grid nodes are classified into supervisor and execute nodes. The authenticator is used to create, execute information database in an execute grid node and remote user information database in a supervisor grid node. User in the execute grid node sends commands to the supervisor grid node to process an operation. If the supervisor verifies and finds it to be correct, it will return ok message to the execute grid node. Users in the execute grid nodes can continue the process through supervisors. In this method, user-id and password are only used to produce authentication that is insufficient to overcome all types of attacks. A novel Authentication Architecture for Grid Security (AAGS) consists of three levels, 0-level node, for root key generation, 1-level node, for sub key generation, and 2-level node, which provides a reliable session connection using trusted link protocol [14]. The AAGS is complex and inconvenient for implementation. The TCP inline authentication is used to enhance network security in grid [15]. In this method, the authentication information is transferred within the TCP three-way-handshake to distinguish the authentication information from application data. The TCP three-way-handshake has been verified for connection depending on the user's the proven identity. This method leads to dynamic operation in firewall. The TCP handshake has been verified before it comes to the final decision. The Identity provided by the user is maintained and stored that is vulnerable for attacks. A unique batch authentication protocol for vehicle-to-grid that reduces authentication delay and less communication traffic compared with the one-by-one authentication scheme has been developed [16]. All parties share a common trust point to have their certificates signed in the scheme which appends the message concatenation in a special format to encrypt or decrypt the message using a public key and a private key respectively. The trust point is vulnerable for attackers. Encryption and decryption are time consuming processes. Reference [8] provides biometric authentication which depends on the password and user-id along with the biometric

data and the geographic position of the user. The same data is used by authorization to reduce the cost and time as well. This scheme optimizes the entry level security and prevents malicious attacks. This method needs special devices which restrict the usage of resource sharing. In general for a grid, user authentication is not sufficient, since a user may get-off and get-on at any time. There is also a possibility that a user may publish user name and password to other uses. The proposed technique identifies the user by a special mechanism of the middleware. The method is fast and does not require special devices for a secure resource sharing.

III. Proposed Work

Security requirements within the grid environment are driven by the need to support scalable, dynamic and distributed virtual organizations [9], [10]. The proposed model consists of three levels such as secure resource accesses, key management and user identity. The first level is managed by Grid Secure Resource Access Protocol (GSRAP) to eliminate the attacks. The inner core 'Key Management' authenticates the user using user name and password that is managed by Re-generation of Irregular Binary Series key holding algorithm (RIBS). The last level is a pure system oriented verification of the user by the unique id of the grid user managed by Grid Resource Identity Protocol (GRIP).

A. GSRAP Registration Mechanism

The GSRAP is a real-time grid platform protocol used to handle the secure resource access in the grid environment. If the users of the grid know that the resource is a protected one, the continuous attacks will be highly possible even by a trusted system. For example if a client on the grid wishes to protect a resource for some limited users, the grid has to face a difficulty which can be resolved using concepts like ontology, Kerberos, etc. In the security issue, the cyclic attack is a major problem, as it will make the file to Denial of Service. If a resource wants to be secured then, the resource administrator should register the resource to the middleware. This involves:

- During registration the resource administrator should submit the private key to the middleware.
- The middleware will encrypt the resource by its private key and give an Access Key File (AKF) for access the resource.
- Similarly the members of the resource should also register with the middleware and added into the AKF.

The resource will be encrypted by the asymmetric key encryption method by the middleware called public key cryptography. In public key cryptography, an asymmetric key pair (a public key and a private key) is used. Normally, the public key is present in the digital certificate issued by the certificate authority. One key (public/private) encrypts the data and the other key decrypts the data. A message encoded with the public key can only be decoded with the private key. The corresponding private keys are secured by the owner and never revealed to the public.

B. GSRAP Resource Access Mechanism

Once the secure registration process is completed by the members with the middleware through the resource administrator then the next step is resource access. If any of the grid users wishes to use the secure resource, then the GSRAP handles the resource accessing system [11], [12]. The following steps are followed.

Step 1. The user has to select the resource.
Step 2. If it is protected, the GSRAP gives an Access Request key (AcReq) to the requester.
Step 3. The AcReq should be submitted to the middleware.
Step 4. The middleware processes it and authenticates the user with the AKF.
Step 5. If authentication is successful, it will check the unique user identity with the access key file. If it is ok, it will give a permission key to the resource and an access key to the resource requester to access the resource.
Step 6. The requester will submit the access key to the GSRAP. Then the GSRAP will permit the resource to access.

IV. Re-Generation of Irregular Binary Series Key Holding Algorithm

The high dynamic nature of a grid computing makes the update of group key(s) a challenging problem. Every participating node would like to offer its resources to be used by other nodes. However, the sharing must be in a controllable and fine-tuned manner. To solve the critical security problems that set obstacle for further deployment and applications of grid computing, this paper presents a mathematical model of a key holding algorithm for an effective communication/sharing of resources in the grid environment. The resource security and non-dependability of other nodes security are focused in this methodology. First, the Unique User Identity (UUID) is presented and followed by Key Cycling mechanism and the User Management is discussed then. This technology could help promoting grid computing to a new era, in which secure resource services offered on the grid are enabled. It is assumed that every valid user/machine in the system is assigned with a Permanent Secret identification key, denoted by PSID. When a user or an organization registers to the grid via the middleware, several certificates need to be issued. The host certificate authenticates the machine involved in the grid. The service certificate authenticates the services offered to the grid. The user certificate authenticates the use of grid services. In this registration process, the permanent secret key can be embedded into the certificates issued to the member.

A. Key Management and Unique User Identity

Each resource in the grid space is identified by a UUID. This UUID is an internal map of the PSID. This service takes place inside the middleware i.e., the UUID is mapped inside the middleware which in turn will take the PSID from each domain and builds mapping table that is represented in Fig. 1. For the mathematical calculation purpose, the UUID is simplified into decimal formats within the range of one to hundred (1-100). To overcome the UUID exceeding from

hundred, a new digit Pi will be included in it and the numbering will start from one. So, the mapping of the resources can be defined as

$$\text{Resource} = P_i.UUID_i \quad (1)$$

The resources are grouped by the middleware according to the limit in each part P_i. This can be stated as

$$P_i = \{UUID_1, UUID_2, UUID_3, \ldots UUID_n\} \quad (2)$$

The Resources (R) in the grid space (G_i) can be termed as

$$R=\sum R_i \quad (3)$$

and the mapping table parts can be represented as follows

$$P = \begin{cases} 1, & \text{if } R \leq 100 \\ \left[\frac{R}{100}\right] + 1, & \text{if } R > 100 \end{cases} \quad (4)$$

In (4), if the Resources (R) is less than or equal to 100, the number parts (P) for the resource mapping is assigned to 1 and if the R is greater than 100, then P will be formulized and assigned. According to the value of the P, the resource mapping table parts will be created.

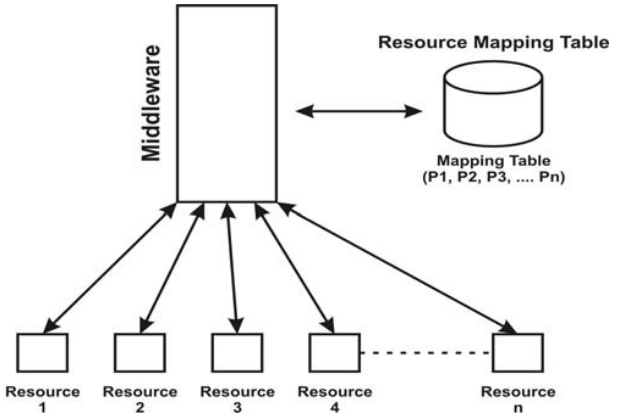

Fig. 1 Resource mapping table with middleware

B. Key Cycling Algorithm and Sin-NOT Series

This is a mathematical method of key generation named 'Regeneration of Irregular Sin-NOT Series (RISS)' method, the Sin-NOT series is Sin (90) =1, Sin (0) =0.

The output of the series is a binary value; these values are taken for the processing by the RISS algorithm. A sample series is shown Fig. 2. The resources are mapped in the sin-NOT series. If the resource exists according to the middleware map, it will be represented as one, and if there is no resource, it will be represented as zero. So, according to the existence of the resources, the Sin-NOT series can be formed and the RISS algorithm generates the digest values. From Fig. 2, the sin-NOT resource series maps resources UUID as 1, 4, 6, 10 and Mapping table part as 1. This series is used in the RISS algorithm to process the key generation. For a secure file, a group or a limited number of users will be allowed by the resource administrator. As the first step user will be given the username and password to authenticate them. This method is introduced in order to overcome the drawbacks of key-servers, ontology concepts and biometric authentication. This includes resource and key management and key regenerative digest methodology.

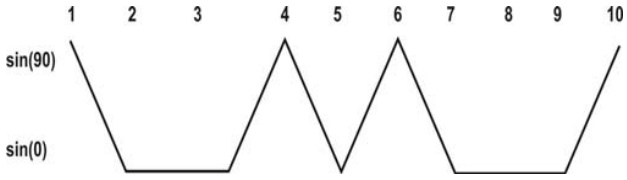

Fig. 2 Irregular sin-NOT series

C. Resource and Key Management

In the middleware, the RISS algorithm and the GRIP plays primary role to manage the resources. The final form of the key can be managed by the secure resource administrator and it can be used by the secure resource. The key can only be read by the middleware, even though the administrator has authenticated with the middleware. If the resource is added or deleted by the administrator, the key file will be re-defined by the middleware and a new key file will be generated for the secure resource. During redefining of the key, the sin-NOT series will be redefined and a new digest value be generated for the existing resource users which will not affect the UUID.

D. Key Syntax

The key is the final part of the RISS algorithm. It is divided into several parts according to the part in (4) representing the number of the resource mapping tables as well as the number of key parts. Each part is capable of holding hundred users within it. Finally in the key, it holds only the revisable hash values of the RISS algorithm or the extraction values of the RISS procedure.

Syntax: <P1, P2, P3… Pn>

<P1 (reversible hash), P2 (reversible hash), P3 (reversible hash), …,Pn(reversible hash)>

Each part is separated by a comma and it represents the end of the part.

V. Generalization of Key Holding Algorithm

An ordinary message digest does not regenerate the series again. However, this method regenerates the series. The representation of output in the form of a single string of digits is created using a formula called a two-way hash function. The sin-NOT series is processed with the mathematical method and the readable output is generated. From the output, the series is generated and the resources are verified which is an electronic authentication. The sin-NOT series is a binary formation which is extracted as binary number i.e., the series consists of the values 0 or 1 in an irregular way. The binary number is converted into decimal numeral system. In Fig. 2, irregular sin-NOT series 1 to 10 is represented in binary form 1001010001. Similarly, a part in P in the middleware will have hundred UUIDs. One key file represents one hundred

resources. These UUIDs will be hashed and the digest value will be stored in the key file. The sin-NOT series binary number is converted to decimal at each level. Let θ_1 be the first digit of the sin-NOT series, θ_2 be the second digit of the sin-NOT series and so on. Therefore initially $\theta_1 = 2^0$,

$$\text{Then } \theta_1 = \theta_1 + \theta_2[\text{next digit varies from 2 to n}]$$

The summation of this series is like the Fibonacci series. Each location is given its decimal equivalent value. The Sin(90) value of the digit is summed with the previous digit starting at 2^0, and evaluating it as "1". Increment the exponent by one for each power. When the amount of elements in the list is equal to the amount of digits in the binary number, it needs to be skipped. For example the number, 10011011, has eight digits. So the list to eight elements would look like: 128, 64, 32, 16, 8, 4, 2, and 1.

Finally θ_1 will consist of the full summation of the sin-NOT series with base ten values.

$$\theta_1 = \theta_1 + 2^{\sin(90)i} \quad (5)$$

Example 1: $2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$

$\theta_1 = \theta_1 + 2^1 = 1+2 = 3$, $\theta_1 = \theta_1 + 2^2 = 3+4 = 7$

$\theta_1 = \theta_1 + 2^3 = 7+8 = 15$, $\theta_1 = \theta_1 + 2^5 = 31+32 = 63$

The above are the base ten equivalents of the sin-NOT series. From these values, the RISS algorithm process gets started. For the re-generation of the sin-NOT series, again the RISS will take input values from the base ten summation values.

fs = 1 (first sum), ls = 63 (last sum)
fls= 64 (first and last sum), ts = 120 (total sum)

The variable ts can be termed as follows,

$$ts = \Sigma_{i=0}^{n} \left(2^{\sin(90)_i}\right) \quad (6)$$

For the re-generative process, the following formula is used to check the sin-NOT series.

Initial difference (df):

$$df = ts - fls \quad (7)$$

$$SVi = \begin{pmatrix} \text{True; } ts=df; fls=2^{\sin(90)i} & , \text{ if}(df < ts) \\ \text{False ; } fls=fls & , \text{Otherwise} \end{pmatrix} \quad (8)$$

Equation (8) is used to re-generate the sin-NOT series again. From this, the middleware can verify whether the UUID of the resource exists in the series or not.

A. Re-generative Process

Initially some base ten values are taken from the series for the re-generative process. From the values, the first value i.e. 2^0 and the last value 2^n values are summed together and kept in a variable fls (first and last summation). Then the total summation is calculated as per (8). Initially, to start the re-generation process, the first and last summation (fls) is subtracted from the total summation (ts). From the second step, the value for the fls is calculated using:

$$fls = 2^{\sin(90)i} \quad (9)$$

After calculating the fls value, (7) and (8) are evaluated. If (8) returns 1 or TRUE, the $2^{\sin(90)i}$ location is marked as the sin-NOT series value else if the equation returns 0 or FALSE, value it is marked as 0 (ZERO). This process is repeated until ts reaches zero (ts=0). By this way, the series gets regenerated and the UUID of the resources are cross checked with the middleware mapping table.

B. UUID vs. sin-NOT Series

In the above example 1, it is shown how the second location of the sin-NOT series is verified. According to the RISS algorithm, the extraction of the sin-NOT series base ten values are as follows,

fs = 1 (first sum), ls = 63 (last sum)
fls= 64 (first and last sum), ts = 120 (total sum)
Using (7), df = ts-fls = 120 – 64 = 56
Using (8), if (df<ts) then SVi=TRUE
ts=ts-df
df=$2^{\sin(90)i}$ else SVi=FALSE
Endif

Since the df is less than the ts value, SVi (series value of location i) consists of binary 1 (ONE) so in the reverse order the last location consist of ONE i.e. (2^5). Then, the target location 2, i.e. to verify this location consisting of ONE or ZERO, again the RISS process begins,

Now, fls = $2^{\sin(90)} = 2^4 = 32$, ts = df = 56
Using (7), df = ts-fls = 56-32=24
Using (8), df is less than ts
∴SVi=TRUE

By the above method, the series is continued to generate as the original series. From this, the middleware can clearly map the UUID with the series and the table.

VI. Algorithm for RISS

The RISS_Base_Ten algorithm is used to convert the binary sin_NOT series to base ten values. The algorithm RISS_Regeneration is used to regenerate the sin_NOT series from the extracted values of the RISS_Base_Ten algorithm.

```
ALGORITM RISS_Base_Ten (sin_NOT_series)
Binary_Number[ ] ←sin_NOT_series;
Series_Length      ←length().sin_NOT_series;
fs =0; // first sum,  ls =0;// last sum
fls =0;//first and last sum,  ts =0;//total sum
for(i=0;i<=Series_Length;i++)
{
 If(Binary_Number [i]==1) then
fs=2(power, (i-1));
If(Binary_Number [i]==n) then
```

```
ls=2(power, (i-1));
If((i>1)and(i<n)) then
sum=sum+2(power, (i-1));
}
 ts=fs+ls+sum;
fls=fs+ls;
END

ALGORITM RISS_Regeneration(fs,ls,fls, ts, series_length)
Binary_Number [ ] ← NULL;
df =0;
for(i=0;i<=series_length;i++)
{
df=ts-fls;
if(fs ≠0) then
{
 if(df<fls)
{
Binary_Number [i]=1;
 ts=df;
}
else
{ Binary_Number [i]=0;
ts=ts;}
fls=2(power, (i-1));
}}
END
```

VII. Results and Discussion

The simulation is made on the GridSim tool kit 5.2, which are real machines connected through internet. The schedule node and database server are also deployed on it. The experimental environment consists of 15 distributed nodes that serve as grid nodes, which are possible to connect 150 different users with different unique identities. The different sizes of resources with authentication times and storage times in the grid environment are shown in Figs. 3 and 4. From these graphs, it is clear that authentication and storage performance variation is small for different resources. The performance of Secure Resource Sharing Protocol with different timelines and different users are represented in Figs. 5 and 6.

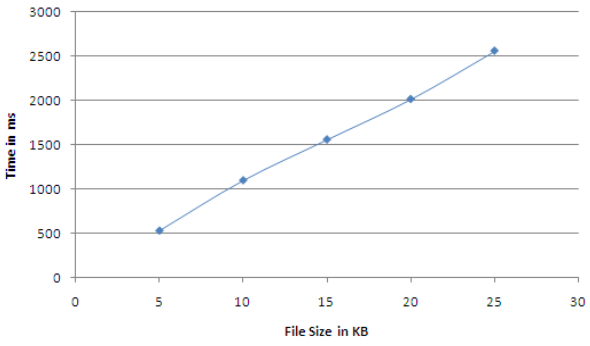

Fig. 3 Authentication vs. Time

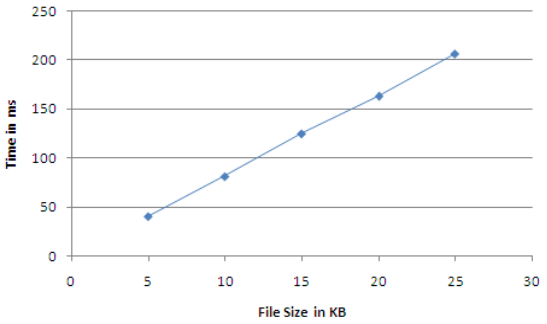

Fig. 4 Storage vs. Time

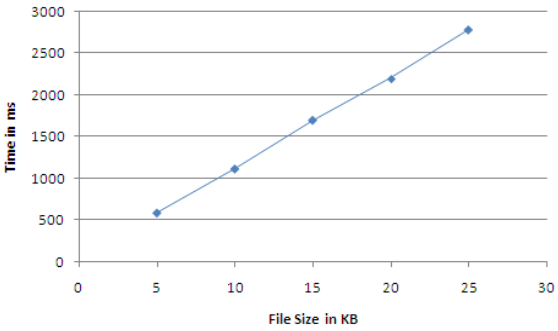

Fig. 5 Secure Resource Sharing vs. Time

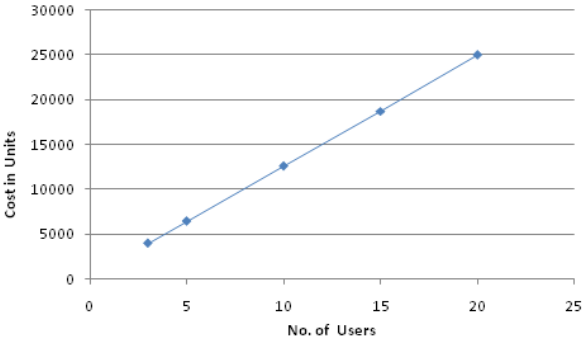

Fig. 6 Secure Resource Sharing vs. Resources users

Fig. 7 shows the different resources processing time of authentication and the UUID mapping. Processing time of different resources is not varying much irrespective of source size. This model is not using any special devices or complex algorithms. Hence, the processing time is not having much variation. Fig. 8 shows the reliability of secure resource sharing. If Reliability is high, it will show that the resources are not easily affected by denial of service. Therefore, the model proposed here is more reliable for grid platform.

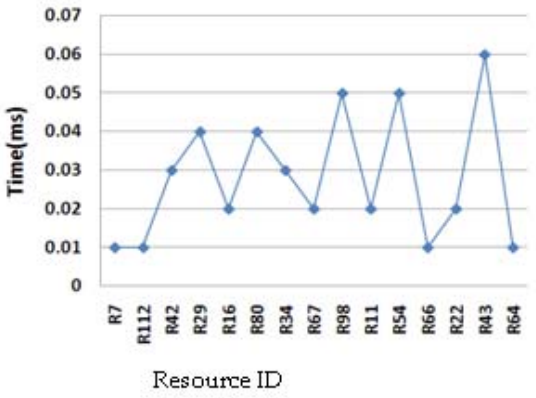

Fig. 7 Secure resource processing time

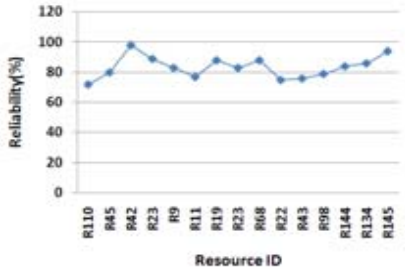

Fig. 8 Resources reliability

Fig. 9 shows the relative performance of various authentication protocols such as Ontology based, Database based and Kerberos based with secure resource sharing. From this graph, it is clear that the proposed algorithm is fast and performance variation is small for different number of resources. This algorithm allows the identification of resources effectively in a simple way.

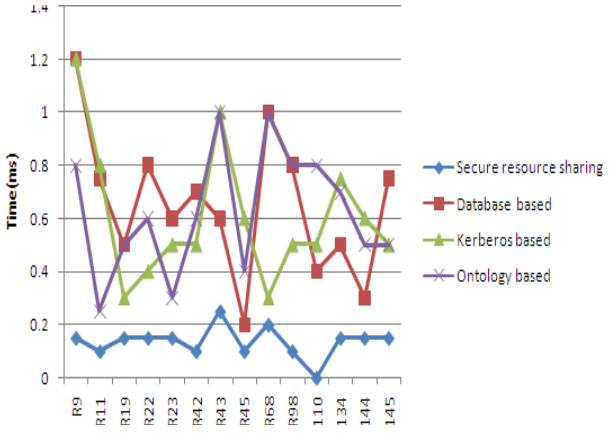

Fig. 9 Secure resource sharing vs. resources performance

VIII. Conclusion

The various authentication protocols were simulated over time and space with different configurations. It supports different application models that can be mapped to resources for the RISS process. The architecture and components of the secure resource sharing mechanism have been discussed. In addition, the average execution times on different resources and middleware have been tested. This result shows that secure resource sharing is fast and this can help to implement real-time security on grids.

References

[1] J. Yu, and R. Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing," J. Grid Computing, vol.3, pp. 171-200, 2005.
[2] B.Allock,, J.Bester, and J.Bresnahan,, "Data management and transfer in high performance computational grid environments, " Parallel Computing, vol. 28, pp. 749-771, 2002.
[3] R.Buyya, and M. Murshed, "GridSim: A toolkit for the modelling and simulation of distributed resource management and scheduling for Grid computing," J. Concurrency and Computation: Practice and Experience, vol. 14, pp. 1061-1074, 2002.
[4] R. Balachandar Amarnath, and Thamarai Selvi Somasundaram," Ontology-based Grid resource Management," *Software Practice and Experience*, vol. 39, pp. 1419–1438, 2009.
[5] I.Blanquer, V. Hernandez, D. Segrelles, and E. Torres, "Enhancing Privacy and Authorization Control Scalability in the Grid through Ontologies," *IEEE Transactions* on information technology in biomedicine, vol. 13, pp. 16-24, 2009.
[6] Tsang-Yean Lee, Huey-Ming Lee, Jin-Shieh Su, and Heng-Sheng Chen, "Processing Authentication Based on Grid Environment," Int. J. Computers, vol. 1, pp. 59-63, 2007.
[7] Downnard Ian., "Public-key cryptography extensions into Kerberos," Potentials *IEEE*, vol.21, pp. 30 – 34, 2003.
[8] G.Jaspher Willsie, and E. Kirubakaran,, "Biometric Authentication and Authorization System for Grid Security, " Int. J. Hybrid Information Tech., vol. 4, pp. 43-58, 2011.
[9] I.Foster, C.Kesselman, J.Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration," *Citesee*, pp 1026-1028.2002.
[10] N.Zhang, L.Yao, A.Nenadic, J.Chin, C. Goble, A. Rector, D. Chadwick, S. Otenko, and Q. Shi, "Achieving fine-grained access control in virtual organizations," Concurrency Computation. : Practice and Experience, vol. 19, pp. 1333– 1352, 2007.
[11] A.Chakrabarti, A.Damodaran, and S.Sengupta, "Grid Computing Security: A Taxonomy," IEEE Security and Privacy, vol. 6, pp. 44-51, 2008.
[12] D.A. Menasce, and E. Casalicchio, "QoS in Grid Computing," IEEE Internet Computing, vol. 8 pp. 85-87, 2004.
[13] A.M. Pernas, and A.R. Dantas Mario, "Using ontology for description of grid resources," Proc. 19th Int. Symposium on High Performance Computing Systems and Applications, 2005, pp 223–229.
[14] Mingyuan Yu, Ronghua Liang, Haibo Yang, and Yahong Hu, "*A Novel Authentication* Architecture for Grid Security," Proc. 4th IEEE Int. Con. on Circuits and Systems for Communications, 2008, pp 95–99.
[15] Jan Wiebelitz, Christopher Kunz, Stefan Piger, and Christian Grimm, *"TCP-AuthN: TCP* Inline Authentication to Enhance Network Security in *Grid Environments*," Proc. 8th IEEE Int. Symposium on Parallel and Distributed Computing, 2009, pp 237–240.
[16] Huaqun, Guo, Yongdong Wu, Hongmei Chen, and Maode Ma, "A Batch Authentication Protocol for V2G Communications," Proc. 4th IEEE Int. Con. on New Technologies, Mobility and Security, 2011, pp 1–5.

M.Victor Jose received his BE degree in Computer Science and Engineering from Manonmaniam Sundaranar University, Tamil Nadu, India in 1995 and ME degree in the same descipline from Madurai Kamaraj University, Tamil Nadu, India in Nov 1996. He is currently pursuing his Ph.D degree at the Department of Computer Science and Engineering, Anna University, India. His research interests include Grid computing, Network security, Data base security, and Multimedia wireless communications.

V.Seenivasagam is working as the Controller of Examinations in National Engineering College (Autonomous), Kovilpatti. He has obtained his B.E Degree in Computer Engineering from Madurai Kamaraj University during the year 1989, M.E and Ph.D degree in Computer Science & Engineering from Manomaniam Sundaranar University in 1997 and 2009 respectively. Dr.V.Seenivasagam has 24 years of experience in teaching and 12 years of experience in research. He has published 16 papers in International journals, 1 paper in National journal, 9 papers in International conferences and 34 papers in National Conferences. He is the recognized supervisor in Anna University of Technology Tirunelveli and Manonmaniam Sundaranar University, Tirunelveli. At present 14 scholars are doing research under his guideship. He has attended many Faculty Development Training Programmes and chaired many international and national conferences and symposiums. He has delivered guest lectures on different topics of Image Processing at various Engineering and Arts Colleges. He has written books on Computer Graphics, Software Engineering, Digital Computer Fundamentals and Fundamentals of Computing and Programming. He is the reviewer for reputed international journals such as International journal on Image and Video Processing, International Journal of Pattern Recognition and Artificial Intelligence, international journal on Association of Modeling Simulation Enterprise and International Journal of Information Technology & Decision Making. He is the member of Board of Studies in the faculty of Computer Science and Engineering at Anna University of Technology, Tirunelveli. His areas of specializations are Image Processing, Compiler Design and Soft Computing,