An Approach to Keep Credentials Secured in Grid Computing Environment for the Safety of Vital Computing Resources

Avijit Bhowmick
Asst. Professor, Department of CSE/IT
Dr.B.C.Roy Engineering College
Durgapur, West Bengal, India

C T Bhunia
Director
National Instituite of Technology
Yupia, Arunachal Pradesh, India

***Abstract*—Presently security attacks have aimed to vulnerabilities in repetitive-use authentication secrets like static passwords. The passwords are used by user in clients side are vulnerable, as the attackers can gain access to a user's password using different types of viruses as it is being typed. These attacks are directing many Grid sites to explore one-time password solutions for authentication in Grid deployment. We present here a novel mechanism called N-LSB where Grid security will be integrated with modified LSB based steganographic technique in order to meet the higher security demands for Grid credentials.**

Keywords- Grid; security; LSB; authentication.

I. INTRODUCTION

Huge network bandwidth, more potent software and hardware of computer at low cost and the flooding of Internet has dived towards need of the latest powerful computing environment. In the late 1990's, Ian Foster et.al. Proposed a complex robust computing environment named Grid computing which was a combination of software and hardware infrastructure where resources are shared for the purpose of computing and storage [1, 2].Because of huge initial investment and high maintenance cost of mainframe or supercomputer, most of the organizations are unable to utilize these type of technologies and again internal resources of the organization also cannot be utilized completely[3].Grid computing technology is defined as "A Computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high and computational capabilities"[4].It refers to the combination of dispersed computer resources from several administrative domains working together to reach a common goal what distinguishes it from conventional high performance systems such as cluster computing [5].

High performance heterogeneous resources like processors, network etc. of different domains are utilized for different Grid applications through Grid middleware. The Globus[12]. Gridbus[15] etc. are the examples of Grid middleware.

Grid computing resources differ from modest number of clusters like the TeraGrid [6] to millions interconnected Pcs like SETI@Home [7].

Since, resources developing an inclusive set of mechanisms and policies for safeguarding the Grid is most essential challenge for Grid computing dubiously in some ways. At present, Grid security analysis and development turns around developing higher solutions to require care of the subsequent requirements: Authentication, Secure data Communication, and effective Security Policies, Authorization, and resource access management. Resources and data security are two basic needs in Grid applications [8, 9]. Coordinated resource sharing and downside breakdown in dynamic, multi-domain virtual organizations are the actual and exact issues that lie behind the Grid conception [10].Intruders attack Grid applications through malicious code, may lead to ruin all the running applications at the same platform. The concept of coordinated resource sharing behind Grid concept is just not only simple file exchange, but also direct access to the vital resources of Grid [11]. The resource administration in Grid environment is difficult due to: (a) geographical dispersion of resources,(b)resource heterogeneousness, (c)having different resource island with their own policies(d) Grid domains inequity [14].

Currently, security is built into Grid computing toolkits like Globus toolkit [12] which is employed at the resource provider sites. The toolkit manages secure channels, authentication [13], resource login, delegation, and resource handling [11]. Among all other issues authentication is the first and most important process of Grid Security Infrastructure (GSI).(GSI) is based on asymmetric cryptography used in a "Public Key Infrastructure" (PKI). Asymmetric cryptography allows users to communicate securely without the necessitate for a previous secret channel to exchange encryption key. Exploiting features of a specific class of mathematical challenges that are simple to generate but virtually impossible to solve (like factorizing large prime numbers), end-entities generate a Complementary set of keys: a "private key" that will be kept secret and a "public key" that is broadcast to the world. Data encrypted with the public key can only be deciphered with the private key (and vice versa). Thus data confidentiality, message integrity and non-repudiation can be attained between two halves of the key pair.

There are many apprehensions about the problems of the current user-managed identity credentials in the Grid PKI. It is essential that the user's private key, encrypted on disk, is

correctly protected both in terms of file protections and in the excellence of the pass-phrase used to encrypt it. As is often the case in the management of user-held Ssh private keys, this does not always happen. The fact that the management of certificates and private keys inside web browsers is also very complex that does not develop the situation.

For this above discussed issue we are going to implement a novel technique developed by us called N-LSB (Nearest Least significant Bit) which is LSB based steganographic advanced modified substitution technique for enhancing security in Grid.

II. LSB Substitution Technique Overview

Substitution based steganography replaces redundant or wasted bits of a digital cover file (digital image, audio or video file) with the bits from the secret message. Let three pixels of RGB scheme as follow:

(00100111 11101001 11001000)
(00100111 11001000 11101001)
(11001000 00100111 11101001)

Changing the first bit i.e. MSB from a 0 (00100111) to a 1 (10100111) will radically change the color; anybody can detect the change with the help of naked eyes. But changing the LSB from 1 (00100111) to a 0 (00100110) cannot be distinguishable in open eyes.

Take a look at how LSB substitution method is applied in steganography to hide a message.

When the character "A" (10000011) is embedded the changed Pixels are

(00100111 11101000 11001000)
(00100110 11001000 11101000)
(11001001 00100111 11101001)

Here for the eight byte information only four have been changed. In best case not a single byte will be changed. But in worst case for eight byte information value of the three pixels will be changed. Sometimes, depending on the pixel, adjusting the LSB can dramatically affect the pixel's properties, making it look out of place in the picture, and therefore subject to detection. This problem can limit the amount of substituted bits, and therefore the size of the secret message. From the above discussion it is clear that keeping the image's properties intact the image size must be eight times greater than the secret message. This is one of the drawbacks of LSB technique. Our N-LSB (Secured-LSB) technique can overcome this problem.

III. Proposed And Developed N-Lsb Method For Keeping Secured In Grid

This new approach not only overcomes the above-mentioned problem but it has an extra layer of security. Like LSB technique N-LSB also require one text file for storing the secret text message and one multimedia carrier file (such as image file, an audio or a video file).

After embedding the source text file information inside each 'atom' of the carrier multimedia cover file, a 'Stego File' is generated whose file type is same as the source cover file. This stego file is used to decrypt the hidden information. N-LSB technique mainly subdivided into two parts: first part, normally done at the sender side is for Encryption of the Source Text File inside the Cover file. The second part, normally done at the receiver side is for Decryption of the secret information from the Stego File.

A. Encryption Method of N-LSB technique:

Step-I

Authenticity checking of the Computer to protect unauthorized use of the source program. This authentication of the system ensures that there will be no misuse of code.

Step-II

Initialization vector is set to the password and is packed into an array.

Step-III

Initialization vector is XOR-ed with the current plain text block to generate the cipher text block & the cipher text stored in a temporary file with a unique file name in the current directory without any kind of user interaction.

Step-IV

This temporary file is actually used to embed the cipher text (stored in it) in a special manner that is based on the old LSB method; but in N-LSB instead of using LSB as the index to embed the text message we generate the index by using Double Hashing Technique.

Step-V

If the source cover file is much smaller than the size of the text information file apply Roll-Back Mechanism to avoid unsuccessful encryption as well as zero probability decryption even if both encryption and decryption password are same.

Here we introduce the variable key generation for the Double Hashing and the value of the key varies depending on the nature of the media files; this does not mean that the key value remains the same for a particular type of media file. The value of the key is different for different files. The 'N-LSB' method actually generates the corresponding indexes that are nearer to the LSB for embedding the text messages. The index may be sometime (LSB-2), sometimes (LSB-1) or sometime LSB itself. If the generated index becomes LSB-2 then LSB-2, LSB-1 and LSB are used, if index becomes LSB-1 then LSB-1 and LSB positions are used, and if the generated index is LSB then only the LSB position is used to embed the secret text information. Using this special feature we try to overcome the above mentioned 1:8 ratio problem.

B. Decryption Method of N-LSB Technique:

Step-I

Authenticity checking of the Computer to protect unauthorized use of the source program. This authentication of the system ensures that there will be no misuse of code.

Step-II

Providing only the correct password (same as encryption password) proper decryption is possible.

Step-III

Using the correct password, the encrypted cipher text is again converted into plain original text in- formation by applying the reverse method already discussed briefly above used to encrypt the secret in- formation. Another exclusive temporary file is used to hold the intermediate results. This temporary file is used to generate the secret information that was sent by the authentic sender.

Step-IV

The two temporary files are deleted as those are no longer needed. We call it Level Zero Destruction.

B. Explanation of the proposed N-LSB Technique :

For each bit of each CIPHER character of the UNIQUE TEMPORARY files, first four bits of the pixel

1. MSB
2. (MSB+1)
3. (MSB+2)
4. (MSB+3)

are added to generate the HASH-KEY hk (for each bit of each CIPHER character of the Unique Temporary file). Clearly (LSB-3) is not involved in calculation in our N-LSB method, we call it a Barrier. It acts as a Barrier between the calculating (MSB, MSB+1, MSB+2, MSB+3) and the calculated (LSB, LSB-1, LSB-2) bits.

Next we follow the following Double Hash Function on each pixel value (for each bit of each CIPHER character of the Unique Temporary file) -

```
int hh=6;
int hash index  hi = (hask key hk % hh ) + 1;
int hm  = (hash key hk % (hh - 2) ) + 1;
int index  = abs( ( (hash index hi + hm) % hh) + 1 );
float ii = (float)index;
```

Finally we check the Ultimate Index (int index) is greater than 5 (Barrier value i.e LSB-3) or not to ensure the Min value of the Ultimate Index (int index) must be 6. If not multiply the hashed index (float ii) with a fractional value 1.55 (i.e. float hmf) and then convert the hashed index (float ii) it to its nearest integer value and stores it to the Ultimate Index (int index).

The above step is continued until Ultimate Index (int index) is greater than the Barrier Value (i.e. 5, i.e. LSB-3).

Another checking is there to ensure that the max value of the Ultimate index (int index) is 8 (i.e. the LSB of that pixel).

One thing is to notice that Ultimate Index (int index) is taken as an Integer while the Hashed Index (float ii) is taken as Float. This is done to calculate the Ultimate Index (int index) more significantly and mathematically eliminating the 0.5 ERROR.

For example if the Hashed Index ii =6.15 the Ultimate Index (index) will be 6 and not 7 but if the Hashed Index ii =6.65 the Ultimate Index (index) will be 7 and not 6.This is called 0.5 Error Elimination. Clearly we have to do the above calculation on each pixel of the Source Cover file for each embedding that is we have to do the above calculation for each bit of each CIPHER character of the Unique Temporary file.

In an Authenticated Machine after checking the Length Error (LE) create a UNIQUE TEMPORARY (UT) file that will contain the Cipher text. UNIQUE in the sense that there should be no name-space collision with any other pre-existing files in the currently working directory and this is assured by the program code. The UNIQUE TEMPORARY file can have any extension. Length Error (LE) occurs if the length of the Text file is larger than the Source Cover file.

We have introduced here a brand-new concept ENCRYPTION PASSWORD where each character of the original text is repeatedly XOR-ed with the characters of the given password to generate the CIPHER text.

For example: one text file contains the data: 1 2 3 4 and user's password is: ab

First 1 is XOR-ed with a

Then 2 is XOR-ed with b

Then 3 is again XOR-ed with a

Ultimately 4 is XOR-ed with b.

From now there is no need of the original text file because now we have got the unique temporary file.

This is done :

a) To remove the dependency factor (of the program on the original source text file).

b) To increase the security level (because we are encrypting the CIPHER text not the ORIGINAL text).

Introducing new concept of password based security where one must know the ENCRYPTION password to decrypt the original text. Until and unless one knows the password given at the time of encryption he is unable to decrypt the original text. Actually, there is no such concept of checking the authenticity of the given password as the old Password concept. If wrong password is entered then decryption will be wrong and the output text is wrong without showing you any ERROR.

Again,in our N-LSB (Nearest List Significant Bit) Method, Double Hashing Technique is used to generate the INDEX (EMBEDDING Location) for each bit of each CIPHER character of the UNIQUE TEMPORARY file. Index can be LSB, (LSB-1) or (LSB-2) of the pixel value.

We follow one of the following rules (i.e., if it satisfies one rule the next rule is not checked):

a. If the Ultimate Index is 6 (i.e., LSB-2) then we replace the 6^{th} (i.e., LSB-2), 7th (i.e., LSB-1) and 8^{th} (i.e, LSB) bit position of the same pixel with the three consequent bits of the Cipher character.

b. If the Ultimate Index is 7 (i.e, LSB-1) then we replace the 7^{th} (i.e., LSB-1) and 8^{th} (i.e., LSB) bit position of the same pixel with the remaining two consequent bits of the Cipher character.

c. If the Ultimate Index is 8 (i.e., LSB itself) then we replace only the 8th (i.e., LSB) bit position of the pixel with only one bit of the Cipher character.

Finally embed each bit of each CIPHER character of the Unique Temporary file in the Ultimate Index or the Double Hashed Index of each pixel of the Source Cover file to generate the Destination File. After the successful Encryption the Unique Temporary (UT) is deleted because it no longer needed. We call it Level Zero Destruction**.**

IV. Performance Analysis And Discussions

Even if Stego file is detected by attacker then also he cannot be able to decrypt the Stego file due to authentication failure because it does not match the computer's environmental specification which was combined with password which adds an extra layer of security. Unfortunately if he gets the correct password then he must have to know the encryption algorithm of N-LSB method which is more secured than so called LSB substitution method which enhances the level of security. If the algorithm is found out one must know the process of variable key generation algorithm using double hashing technique. It is quite difficult to find out the hash functions that have been used.

LSB Technique	Proposed N-LSB Technique
Exists 1:8 ratio problems.	Removed 1:8 ratio problems.
Only LSBs are used, so less information can be embedded.	Nearest bits of LSBs are also utilized, so in same cover file more information embedded.
Plain text information is directly kept in LSBs, less secured.	Plain text encrypted first, then kept in nearest LSBs, complete security for data.
For maximum information embedding in cover file, all the LSBs are utilized.	For the same amount of information embedding in same cover file 37.5 % less LSBs are needed.

Fig.1 Performance comparison

Again, in LSB substitution technique to embed a character eight bytes must be extracted from the cover file. But in N-LSB method eight bytes needed in worst case if and only if the generated index is only LSB in all cases. In best cases, if it produce index LSB-2 then only three bytes will be extracted from the cover file. So in best case, it requires only 37.5% of the cover file compared so we can increase the embedding capacity more than 60%. Obviously in average case, N-LSB method requires 50% less size of cover file and it increase 50% embedding capacity.

We can also increase the embedding capacity by converting the entire secret message into either lower case or in uppercase. With these multilevel of security the proposed N-LSB method is secured from attacks than any other existing techniques.

V. Conclusion

Vital information through LSB technique reduces the probability of detection of the information unless knowing the cryptographic algorithm behind the mechanism. Utilizing this unique feature, we have implemented through our developed unique method called N-LSB Technique where we have observed less time and less no of bits are needed for keeping users credentials secured. So, this robust technique is very much useful to keep Grid credentials secured for ensuring data integrity and information security in Grid computing environment for issue of accessing vital heterogeneous resources in secured manner.

References

[1] F. Berman, G. Fox and T. Hey (eds.), Grid Computing: Making the Global Infrastructure a Reality. Wiley, 2003.

[2] M. Cosnard and A. Merzky, "Meta- and Grid-Computing", in Proceedings of the 8th International Euro-Par Conference, August 2002, pp. 861–862.

[3] Nadia Ranaldo, Eugenio Zimeo. A Framework for QoS- based Resource Brokering in Grid Computing. In 5th IEEE ECOWS, the 2nd Workshop on Emerging Web Services Technology, Halle, Germany, 2007.

[4] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the Grid: Enabling scalable virtual organizations." Int. J. Supercomputing, vol. 15, no. 3, pp. 200-222, 2001.

[5] Foster, I. and Kesselman, C. Computational Grids. Foster, I. and Kesselman, C. eds. The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, 1999, 2-48.

[6] "National Science Foundation TeraGrid". From http://www.teraGrid.org.

[7] SETI@Home: The Search for Extraterrestrial Intelligence. http://setiathome.ssl. berkeley.edu/

[8] F. Berman, R. Wolski, H. Casanova, W. Cirne, H. Dail, M. Faerman, S. Figueira, J. Hayes, G. Obertelli, J. Schopf, G. Shao, S. Smallen, N. Spring, A. Su and D. Zagorodnov, "Adaptive Computing on the Grid Using AppLeS", IEEE Trans. on Parallel and Distributed Systems, Vol. 14, April 2003.

[9] V.Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S.Meder, L. Pearlman and S. Tuecke, "Security for Grid Services", in Proceedings of the HPDC-12, 2003.

[10] Zhiguo Shi, Yeping He, Xiaoyong Huai, Hong Zhang. Identity Anonymity for Grid Computing Coordination based on Trusted Computing. Proceedings of the Sixth International Conference on Grid and Cooperative Computing. pp. 403-410, 2007.

[11] Foster, I., Kesselman, C., Tsudik, G. and Tuecke, S. A Security Architecture for Computational Grids. ACM Conference on Computers and Security, 1998, pp: 83-91.

[12] I. Foster. Globus toolkit version 4: Software for service- oriented systems. In Proc. of the IFIP International Conference on Network and Parallel Computing, 2005.

[13] J. Basney, W. Nejdl, D. Olmedilla, V. Welch, and M. Winslett. Negotiating trust on the Grid. In 2nd Workshop on Semantics in P2P and Grid Computing, New York, May 2004.

[14] Farag Azzedin, Muthucumaru Maheswaran, "Towards Trust-Aware Resource Management in Grid Computing Systems," ccGrid, p. 452, 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02), 2002.