Indian Journal of Science and Technology, *Vol 8(15), IPL049, July 2015*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Web Service Authentication and Multilevel Security

M. Thiyagarajan[1*], Chaitanya Raveendra[2] and V. Thiagarasu[3]

[1]Nehru Group of Institution, Nehru Gardens, Thirumalayampalayam, Coimbatore - 641 105, Tamil Nadu, India; m_thiyagarajan@yahoo.com
[2]Department of Computer Science, Karpagam University, Nehru Group of Institutions, 451-D, Kuniamuthur, Coimabtore - 641 008, Tamil Nadu, India; chaitanya2575@gmail.com
[3]Computer Science, Gobi Arts and Science College, Gobichettipalayam - 638 453, Tamil Nadu, India; chaitanya2575@gmail.com

Abstract

Objective: The objective of the work is the implementation of trusted system in network security. The problem has been identified as minimizing the time and space complexity while adding the vertices and edges in the implementation of zero knowledge protocol using Graph Isomorphism, the identification of the classes of the graph and the distribution of the isomorphic graphs as the private keys in public key cryptosystems for authentication. **Methods:** Guillou-Quisquarter algorithm is used as basic zero knowledge protocol. An adjacency matrix is generated by reordering the vertices and communicated for authentication. Constructed the isomorphic graph using the nauty algorithm and classify this to each set of users. Nauty algorithm is used to check for the isomorphism of graphs which uses the canonical method using partitions and search tree. Implemented the partition algorithm and generated the search tree. Three graphs are generated as sample data. We have used the toy example of Bank Loan to implement the new method. **Findings:** The improvised algorithm minimizes the time and space complexity. We have increased the vertices and proved that the time complexity won't increase with the increase in the number of vertices. We have identified the classes of graph for each set of users for authentication. This graph is given as the key in public key cryptosystems for the implementation of trusted system. Isomorphic graph is generated as the key for each set of users to implement the multilevel security. Thus the authentication and the multilevel security aspects have been handled by the use of class of isomorphic graphs and Guillou Quisquarter protocol. **Application:** We have identified the area of application in the implementation of I/O automata, finite state machines and timed machines for authenticating against the set of resources.

Keywords: Bank Loan, Graph Isomorphism, Nauty Algorithm, Partition Algorithm, Zero-Knowledge Protocol

1. Introduction

William Stallings[1] in Cryptography and Network Security states that Security is a concern of organizations with assets that are controlled by computer systems. By accessing or altering data, an attacker can steal tangible assets or lead organization to take actions it would not otherwise take. By merely examining data, an attacker can gain a competitive a competitive advantage, without the owner of the data being any the wiser. System Security can be implemented in three different aspects based on different type of threats and its implementation strategies. Intruders attacking the system have been protected using intruder detection system and password management system. Software threats from different software like virus, worms can be handled with the help of Antivirus programs. The third and final level of security can be handled using firewall and trusted system. Firewall acts as a barrier and will accept the traffic if it has the authorized data. Trusted system implements the authentication of the file a user handles. It is mainly handled using access control using access matrix or an access control list.

A significant security problem for networked system is the trespass or unwanted access by users and software. Users trespass can be termed for unauthorized log on to the machine, or the unauthorized privilege access of

**Author for correspondence*

an authorized user. All these attacks relate to network security as the communication is possible through network. Many types of intruders can be handled using Audit records, Statistical Anomaly detection, Rule based Intrusion detection, Base Rate fallacy, Distributed Intrusion detection, honeypots and password protection. Firewall can be implemented using packet-filtering router, Stateful Inspection Firewalls, circuit level gateway and bastion host.

The most efficient way to handle the intruders and the malicious software is the implementation of the trusted system technology. Trusted system deals with the protection of data or resources based on the basis of the permission of the user. This when implemented in the network can be called as trusted network security. Thus when applied in the web, we are having a set of users and the set of data that can be accessed by the users. In our paper, we implemented a trusted system which has a set of users having access to a particular set of resources. We have used zero knowledge using an extended graph isomorphism algorithm to implement the security.

Eric ayah[2] described on the common problems faced by the traditional system as impersonation. Impersonation can be stated as the act of imitating ones actions, like listening to ones communication, collects information so that he can use the information gained to access the system. There are several solutions available to cover this problem. Secret key cryptosystems involves the use of secret key for the exchange of data. Another approach is the public key cryptosystems where the data is send using the public key and he must generate the output using the private key. But, both the protocols described above suffer MITM (Man in the Middle Attack).

An alternative to the MITM attack and the traditional password management is the use of zero knowledge protocols. ZKP is an interactive protocol where a prover can prove his worth using some mathematical methods such as quadratic residues, graph-3 Color ability, prime factoring, Hamiltonian Cycles for large graphs, and graph Isomorphism. In the paper titled Trusted System, we have implemented the zero Knowledge along with Graph Isomorphism[3].

2. Basic Concepts and Results

This section covers the definitions which are used in the current paper. Here in this paper we implemented the trusted system for the authentication process using network protocol. Whole transaction or communication has been carried by the continuous exchange of data. The data transferred is highly sensitive like credit card password, bank account details etc. which when reached an unauthorized person is a greatest flaw in the communication. Loss once incurred cannot be roll back.

2.1 Definitions

2.1.1 Multilevel Security

The multilevel security is defined as a subject at high level may not convey information to a subject to a lower or a non comparable level unless that flow accurately reflects the will of an authorized user. A multilevel secure system must enforce simple security property and star property. These two rules, if properly enforced, provide multilevel security. It implies that security is provided to multiple categories or multiple levels of data.

2.1.2 Access Matrix

A general model of access control as exercised by a file or database management system is that of an access matrix. The basic elements of the model are subject, object and access right. One axis of the matrix consists of user or any application or any processes. The other axis represents the file, portion of a file, programs or a data.

2.1.3 Graph Isomorphism

Two graphs which contain the same number of graph vertices connected in the same way are said to be isomorphic. Formally, two graphs G and H with graph vertices $V_n = \{1, 2, \ldots, n\}$ are said to be isomorphic if there is a permutation P of V_n such that $\{u, v\}$ is in the set of graph edges $E(G)$ iff $\{p(u), p(v)\}$ is in the set of graph edges $E(H)$. Canonical labeling is a practically effective technique used for determining graph isomorphism[4].

A graph G is a set of nodes or vertices V, connected by a set of edges E. The sets of vertices and edges are finites. A graph with n vertices will have: V = {1, 2, 3,..., n} and E a 2-element subsets of V. Let u and v be two vertices of a

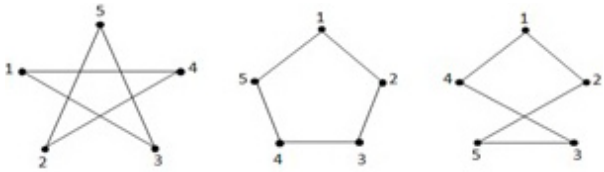

Figure 1. Isomorphic Graphs.

graph. If (u,v) ε E, then u and v are said to be adjacent or neighbors. A graph is represented by its adjacency matrix. For instance, a graph with n vertices, is represented by a n×n matrix, where the entry is "1" if there is an edge linking the vertex i to the vertex j, and is "0" otherwise. For undirected graphs, the adjacency matrix is symmetric around the diagonal. Example is shown below[14].

2.1.4 Zero Knowledge

ZKP is an interactive protocol where a prover can prove the veracity of a statement to a verifier without disclosing any other information, which could allow an eavesdropper or the verifier to impersonate him. During a ZKP interaction, the prover will try for example to convince the verifier that he/she knows the secret password to open a door without actually giving the password to the verifier. The verifier throughout the interactions will ask questions in the aim of verifying that the prover really knows the secret password. If the answer to a question is wrong, the communication is immediately terminated, or the access to the network is denied.

2.1.5 Miyazaki Constructions

In his paper "The complexity of McKay's canonical labeling algorithm," Miyazaki constructed regular graphs based on the Cai-Furer-Innerman construction, which he proved to be very complex for nauty, therefore for canonical labeling algorithms in general. Miyazaki constructed a family of colored graphs which force nauty to run in exponential time. Miyazaki's construction demonstrates that the coloring significantly affects the behavior of the algorithm, and can turn out to be the difference between polynomial and exponential runtime. Figure 2 shows an example of miyazaki constructions[15].

2.2 Proposition

2.2.1 Nauty Algorithm

The most widely used of canonical labeling programs is nauty by Brendan McKay. Canonical labeling is a practically effective technique used for determining graph isomorphism. Most canonical labeling programs such as nauty (no auto morphisms yes), use a backtracking algorithm that goes through the search tree in the aim of finding a canonical label, while building the auto morphism group of the graph. There are three main strands to the nauty algorithm:

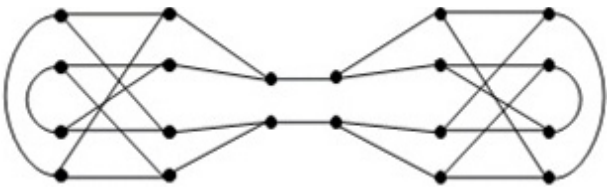

Figure 2. Miyazaki Graphs.

1) Using, iteratively, degree information;
2) Building a search tree examining choices not determined by degree information; and
3) Using graph auto morphisms, as they are found, to prune the search tree.

Nauty algorithm has been regarded as the fastest isomorphism testing algorithm. Other algorithms that offering a better performance are bliss of TommiJuntilla and PetteriKashi, saucy of Martin Kutz and Pascal Schweitzer, sinauto and conautoof Jose Luis LopezPresa, Traces of AldofoPiperno. The main reason behind the impressive performance that nauty has been offering are:

1) The reduction of the graph isomorphism problem to the problem of finding a canonical label for each of the graph being compared,
2) The use of auto morphisms found while searching for a certificate to prune the search tree, and
3) The use of invariants.

In order to find the canonical label of a graph, nauty starts with the initial partition based on the degrees of the vertices and generates a search tree. The initial partition or root of the search tree is considered to be at level 0. Since the canonical label is generated from the smallest of the auto morphs, nauty performs comparison on all possible leaf partitions of the search tree. Algorithm determines the target cell which is most likely to contain the greatest number of vertices. Once such cell is selected, nauty process to individualization of each vertex of the cell in order to refine the partition. This is done until a leaf partition is reached. When a leaf partition is reached, its corresponding canonical label is computed and stored as a potential canonical label for the graph. The nauty program then backtracks to the parent of such partition at an upper level in the tree.

From this ancestor, another leaf partition is generated. The canonical label associated with the new leaf partition

is computed and compared with the previous one. If the value is the same, an auto morphism is discovered and saved. On the other hand, if this new value is greater than the previous, it is automatically discarded. If conversely the new value is better, it will substitute the old label as the new potential certificate. Subsequently, the auto morphism found is used to prune the search tree. The process is repeated until a canonical label is obtained for the graph.

2.2.2 Partition Algorithm

A partition $\Pi=V_1, V_m$, divides the vertices of a graph into non-empty subsets of V which are called cells. In order to process to the refinement, algorithm starts with an initial partition, where the vertices in each cell have the same degree. It then selects the first cell with more than one element namely the target cell, and computes a sequence a_v based on the adjacencies of the vertices in the target cell with the nodes in all the cells of the initial partition. Then, it uses the calculated sequence to split the target cell into a number of subsets, so that the vertices in each subset have the same value for a_v. The following describes the partition for the algorithm.

```
Function partitionAlg()
{
String π[] = {V1,.. … , Vm};
For each(Vi in π[])
   {
If (Vi has more than one vertices)
        {
av = (d(v, V1), …, d(v, Vm));
}
   If (getSubset(Vi) == av)
      {
         Return value
       }   }}
```

2.2.3 Search Tree

McKay's algorithm starts by forming the equitable refinement of the unit partition, thereby extracting all of the initial degree information. Having reached an equitable partition, we need to introduce artificial distinctions between vertices. However, we must be careful to examine all relevant choices. We systematically explore the space of equitable ordered partitions using a search tree. The next definition describes the way we make these artificial distinctions, forming children in the search tree. Let π be an equitable ordered partition of [n] with a nontrivial part Vi, and let u ∈ Vi. The splitting of π by u, denoted by $\pi \perp u$, is the equitable re-finement R (π′) of the ordered partition $\pi' = (V_1, V_2, \ldots, \{u\}, V_i \setminus \{u\}, V_{i+1}, \ldots, Vr)$. (Note that $\pi \perp u$ is strictly finer than π.). Example of a search tree generation is shown if Figure 3[16].

3. Authentication Models based on Zero Knowledge

The following sections describe the implementation of the authentication with the using of Zero Knowledge Protocol, the isomorphism and the Nauty Algorithm. The first section deals with the first part of Zero Knowledge Protocol. The second section deals with the Implementation of Graph Isomorphism based on relabeling of vertices. The final section deals with the graph isomorphism using Canonical labeling algorithm.

3.1 Guillou-Quisquarter Algorithm

One of the most important, and at the same time very counter intuitive, primitives for cryptographic protocols are called zero knowledge proof protocols[7–13]. Informally, a zero knowledge proof protocol allows one party, usually called prover, to convince another party, called verifier, that prover knows some fact (a secret) without revealing to the verifier any information about the college. We have used Guillou-Quisquarter algorithm to implement

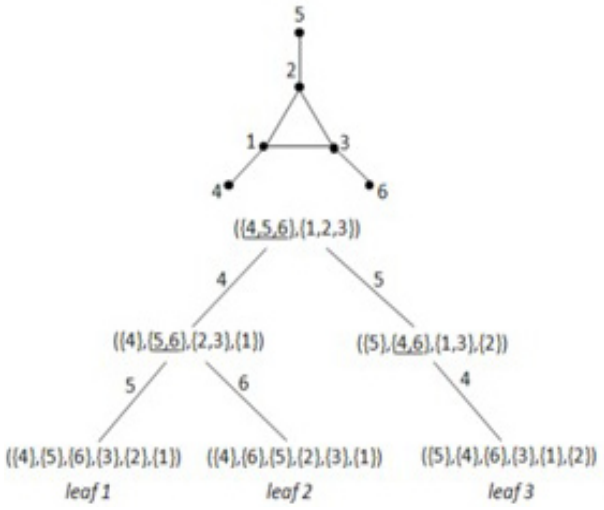

Figure 3. Search Tree Generation.

the zero knowledge protocols. Following describes the working of the Guillou Quisquarter algorithm.

1. Generate a random number r.
2. Prover sends $X=r^v \bmod n$.
3. Verifier sends e from 1 to n-1.
4. Prover reply with $Y=rs_a^e \bmod n$
5. Verifier then computes,
6. $J_A=f(I_A)$.
7. $Z=J_A^e * Y^V \bmod n$.
8. If Z=X Access Granted
9. If Z=0 Access Denied

The authentication process discussed demands the computation of large primes and their powers. Thus, a small addition to the zero knowledge interactive protocol which minimizes the delay in computational process has been identified. The number theoretical competition can be minimized through graph isomorphism algorithm and zero knowledge proofs.

3.2 Graph Isomorphism based on Relabeling of Vertices

Two graphs *G1* and *G2* are said to be isomorphic, if a one-to-one permutation or mapping exists between the set of vertices of *G1* and the set of vertices of G2, with the property that if two nodes of *G1* are adjacent, so are their images in *G2*. The graph isomorphism problem is therefore the problem of determining whether two given graphs are isomorphic. Suppose there are two graphs *G1* and *G2*, such that the graph *G2* is generated by relabeling the vertices of *G1* according to a secret permutation π while preserving the edges. The pair of graphs *G1* and *G2* forms the public key pair, and the permutation π serves as the private key. A third graph *H*, which is either obtained from *G1* or *G2* using another random permutation, say ρ is sent to the verifier who will in return challenge the prover to provide the permutation σ which can map *H* back to either *G1* or *G2*.

Given *G1* and *G2* such that $G2=\pi(G1)$, the interactions constituting a round of the graph isomorphism based ZKP protocol is illustrated as follows:

1. Prover chooses randomly selects a $\varepsilon \{0, 1\}$
2. Prover chooses a random permutation ρ, and generates $\rho(G_a)$
3. Prover sends the adjacency matrix of *H* to the verifier.
4. Verifier sends $b \varepsilon \{0, 1\}$ to the prover and challenges for σ which maps *H* to *Gb*.
5. If a = b the prover $\sigma = \rho^{-1}$ sends to the verifier.
6. If a = 0 and b = 1 the prover sends $\sigma = \rho^{-1} \circ \pi$ to the verifier.
7. If a =1 and b = 0 the prover sends $\sigma = \rho^{-1} \circ \pi^{-1}$ to the verifier.
8. Verifier checks if $\sigma(H) = G_b$ and grants access to the prover accordingly

3.3 Graph Labeling based on Nauty Algorithm

In the first approach we develop an adjacency matrix to represent the graph and compare to check for the consistency. And that algorithm generates and transports an adjacency matrix to and fro for the interaction. As zero knowledge protocol is an interactive protocol, both the client and the server are communicating using the adjacency matrix which consumes the space and time of the system. Thus we reach a new conclusion to just send a permuted graph and check whether a user is able to generate the original graph. If he is able to produce, then granted .otherwise it will check for another set of iterations. Thus the modified program for zero knowledge with graph isomorphism can be re-written as

1. Generate 3 public graphs G1, G2 and G3.
2. Π determines the permutations of the graphs.
3. When an access request came, server will generate a permuted value of Graph G2 or G3; if client is able to generate the Graph he is given access.
4. Each iterations check for true or false.

Thus this new algorithm minimizes the time complexity. The structure of the graphs plays an important role in strengthening the algorithm. During an interaction of the zero-knowledge protocol or permutations is sent to the verifier by the prover. In the case an eavesdropper is listening to the conversation and is able to intercept, it would be easy for him to impersonate the prover if and only if, the graph used is not hard for the isomorphism problem .A simple invariant that all practical graph isomorphism algorithms use when solving the isomorphism,

or automorphism problem, is the degree of the vertices constituting the graphs.

There are so many graphs which are used along with the canonical algorithm. The graphs are Projective Planes (PP), Cai-Furer-Immerman construction, Constraint satisfaction problems, Hadamard matrices (Had) , Miyazaki's constructions (Mz), Affine and projective geometries, Random regular graphs (Rnd) , Strongly regular graphs and Grid graphs. Also, unions of graphs with similar structures are known to be very complex for the nauty program. Miyazaki constructed regular graphs based on the Cai-Furer-Innerman construction, which he proved to be very complex for nauty, therefore for canonical labeling algorithms in general. Miyazaki graphs works best for nauty.

4. Illustration with an Example: Bank Loan

Here, in this paper we have implemented the trusted security in the web service. Trusted system ensures the security for the users against the set of resources. In this web scenario, we have a set of customers and resources, so here we have allocated the resources to n number of resources using token algorithm. Now we have to implement the trusted property using the zero knowledge with the graph isomorphism. We have created two web services; categorization and authentication. When a user enters he will grouped in to a particular level which takes care of the multilevel security. We have categorized the user based on his criteria, if he is able to generate a graph and got the access and based on the category the loan will be sanctioned. If a user belongs to category1 can retrieve the amount of Rs. 1000/- to the account, but not more than that. But a user belonging to category2 can retrieve up to Rs. 5000/-

4.1 Sample Data

We have used three graphs g1, g2 and g3 as public keys. User is given the secret key to access the money. When a user log on to the system, he will get the category. The server will send a permutated graph to the customer, if the customer is able to construct a graph g2 in all iterations, he will be given access. In the sample program we have used 6 vertices to implement the graph. Secret keys are permutations. The secret key of g1 is "152436" and the secret key of g2 is "135426"

4.2 Screenshots

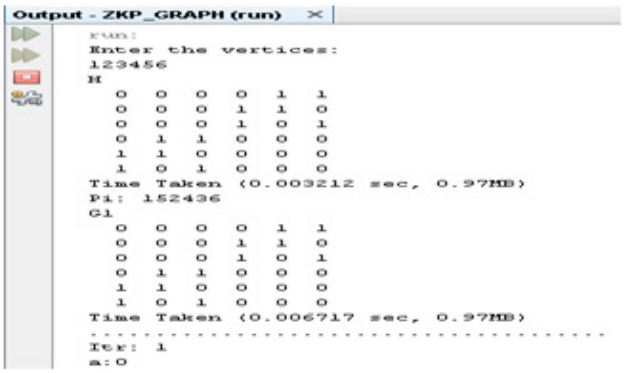

Figure 3. Screenshot of Inputs.

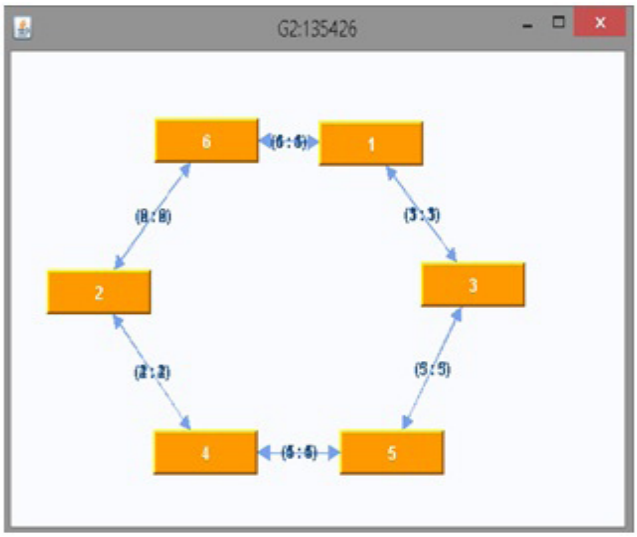

Figure 4. Screenshot of Output.

4.3 Result and Discussions

This system is constructed for two services, one for user categorization and other for authentication. The system contains a set of users with a set of resources. For sample illustration we consider an example of bank loan. Here, one user category can be able to withdraw $1000 and other category will be able to withdraw $2500.When a user log on to the system, he is categorized. We constructed a two isomeric graph g2 and g3. If logged user can be able to generate isomeric graph, then according to his category transaction will occur. The current scenario uses six vertices graph and we executed the system by increasing number of vertices of a graph at 1000. We can prove that the execution time can be reduced to nana second at controlled lab conditions. Server generates a random graph

Table 1. Time Complexity

No of vertices	Time Complexity
5	0.013199 sec
6	0.013171sec
7	0.015724 sec
9	0.017612 sec

152436, after the executions, it generate back the original graph g2, he is given access. If he is not able to generate, then no transaction is allowed. We can generalize the problem to m customers to n rights or the combination of n rights. We can increase the number of vertices, as the increase of vertex after 400 will make the intruders difficult to break the data, thus the access will be finished within the time limit. Graph isomeric using Nauty algorithm is best chance to implement. We can generalize the services for "m" customers for "n" resources. The following table describes the time complexities when number of vertices is increased.

5. Conclusion

To enhance the ability of the system to defend against intruders and malicious programs, the best way is to implement the trusted system technology. This in turn give rise to different access rights which is being exercised by users in series and parallel. We addressed a problem and eliminated the thread anticipated by the server and clients in transmitting data over web for service we find our attempts to better results than other protocols employed and quick implementation to overcome the issues. Thus created a sample program to illustrate the working and implemented the same.

In the future, we can increase the number of vertices used in the algorithm and also making it strong and tough using the hard graphs. The categorizations and resources can be generalized to accommodate any problem, thus as a generalized service to any request. We have to give specifications for all the set of customers and resources thus enforcing the network security in a distributed environment.

6. References

1. Stallings W. Cryptography and Network Security. 4th Edition. USA: Pearson Education; 2002.
2. Ayeh Eric. An investigation into graph isomorphism based zero-knowledge proofs (MS Thesis). University of North Texas; 2009 Dec. Available from: http://nsl.cse.unt.edu/dantu/cae/attachments/Eric_Ayeh_MS_thesis.pdf
3. Wang H, Sun Z, Nanjing C. Research on zero- knowledge proof protocol. IJCSI; 2013 Jan; 10(1):194–200.
4. Karpagaselvi S. Soft computing Techniques for web search Engines [Unpublished PhD Thesis]. Chennai: Anna University of Technology; 2012 Jun.
5. Bayer S, Jens G. Efficient zero-knowledge argument for correctness of a shuffle. EUROCRYPT; 2012. p. 263–80.
6. Chaitanya R, Thiyagarajan M. Trusted System. International Journal of Computing Algorithm. 2015 Mar; 04:1303–06. ISSN: 2278-2397.
7. Garg S, Jain A, Sahai A. Leakage-resilient zero knowledge. CRYPTO; 2011. p. 297–15.
8. Gupta A, Stahl D O, Whinstone A B. Managing computing resources in intranets: an electronic commerce perspective. Decision Support System; 1998; 24:55–69.
9. Goldreich O, Micali S, Wigderson A. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design, J FOCS; 1986. p. 174–87.
10. Grzonkowski S, McDaniel Z. Extending web application with a lightweight zero knowledge proof authentication. Proceedings of the 5th international conference on Soft computing as trans disciplinary science and technology (CSTST '08); ACM; 2008 Oct. p. 65–70.
11. Lin H, Pass R, Tseng W-LD. Muthuramkrishnan Venkitasubramaniam: Concurrent Nonmalleable Zero knowledge proofs. 30th Annual International cryptology conference, CRYPTO; 2010. p. 429–46.
12. Udgata SK, Mubeen A, Sabat SL. Wireless sensor network security model using Zero Knowledge Protocol. Proceedings of IEEE Communication (ICC); 2011. p. 1–5.
13. Lin W-L. Concurrent Non-malleable Zero knowledge proofs. 30th Annual International cryptology conference, CRYPTO; 2010. p. 429–46.
14. Eric A. An Investigation into Graph Isomorphism Based Zero-Knowledge Proofs [MS Thesis]. University Of North Texas; 2009 Dec. Available from: http://nsl.cse.unt.edu/dantu/cae/attachments/Eric_Ayeh_MS_thesis.pdf, Figure.1, Example of isomorphic graphs with their corresponding adjacency matrices; p. 10.
15. Eric A. An Investigation into Graph Isomorphism Based Zero-Knowledge Proofs [MS Thesis]. University of North Texas; 2009 Dec. Available from: http://nsl.cse.unt.edu/dantu/cae/attachments/Eric_Ayeh_MS_thesis.pdf, Figure 2, Example of Miyazaki's graph; p. 32.
16. Eric A. An Investigation into Graph Isomorphism Based Zero-Knowledge Proofs [MS Thesis]. University of North Texas; 2009 Dec. Available from: http://nsl.cse.unt.edu/dantu/cae/attachments/Eric_Ayeh_MS_thesis.pdf, Figure 3, Example of search tree; p. 23.