2009

# Chinese Wall Security Policy

Varun Gupta
*San Jose State University*

# Chinese Wall Security Policy

A Project

Presented to

The Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Masters of Science

By

Varun Gupta

December 2009

SAN JOSÉ STATE UNIVERSITY

The Undersigned Project Committee Approves the Project Titled

CHINESE WALL SECURITY POLICY

by
Varun Gupta

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

Prof. T.Y Lin, Department of Computer Science Date

Prof. Chris Pollett, Department of Computer Science Date

Prof. Robert Chun, Department of Computer Science Date

APPROVED FOR THE UNIVERSITY

Associate Dean Date

ABSTRACT

Chinese Wall Security Policy

by Varun Gupta

*This project establishes a Chinese wall security policy model in the environment of cloud computing. In 1988 Brewer and Nash proposed a very nice commercial security policy in British financial world. Though the policy was well accepted, but the model was incorrect. A decade later, Dr. Lin provided a model in 2003 that meets Brewer & Nash's Policy. One of the important components in Cloud computing is data center. In order for any company to store data in the center, a trustable security policy model is a must; Chinese wall security policy model will provide this assurance. The heart of the Chinese Wall Security Policy Model is the concept of Conflict of Interest (COI). The concept can be modeled by an anti-reflexive, symmetric and transitive binary relation. In this project, by extending Dr. Lin's Model, we explore the security issues in the environment of cloud computing and develop a small system of the Chinese Wall Security Model.*

ACKNOWLEDGEMENT

First, I would like to thank my master's advisor Dr. Tsau Young Lin for his invaluable insight and inspiring guidance, which worked well towards my motivation to work and research for this topic.

I would also like to thank my committee members for helping me improve this project.

Finally, I would like to express my sincere thanks to the Department of Computer Science at San Jose State University to provide me such an opportunity to explore and publish my views and ideas for my topic.

**Table of Contents**

# List of Figures

## 1. Overview

This chapter gives an introduction to Cloud computing & Chinese wall security policy and also provides a detailed definition with example to explain the concept of Chinese Wall.

### 1.1 Introduction

This project provides security to the cloud users by creating a wall between competing companies. It will provide users with a secure system called Chinese Wall Secure Discretionary Access Control (CW-DAC). In other words, at the completion of the project, there will be a secure system which will let the users know that whether their information flow is secure or not. Basically, Cloud computing is a process which provides hosted services, like Software as a Service, over the Internet in a fast and cost-effective way. This technology gained popularity in a little span of time as the companies are always looking for a way to save money, but as always, this technology also has some risks, and it could have the company left with security vulnerabilities and threats since all the computation is done on data center. Basically cloud computing and Chinese wall come hand in hand. Chinese Wall Security Policy can be most easily visualized as the code of practice that must be followed by a market analyst working for a financial institution providing corporate business services [Bre89]. So the analyst cannot provide any information of his clients to the competing companies or the status and plans of the company. But the analyst can always share the information with the companies that are not in competition with their client's companies. In other words, there is no problem in sharing information with the company's friends (assuming friends form an equivalence

relation). Companies always want to confirm that, no matter where is there data, it is protected. CW-DAC Model provides security to the cloud users by grouping all the users who are friends (friendship is assumed to be an equivalence relation) in one single class and enemies in another class, known as Conflict of Interest Class, which is anti-reflexive , symmetric and antir-transitive. Many instances of Chinese Walls can be found in the financial world. The data access depends upon what data the company already has access to. So the datasets are actually grouped into Conflict of Interest classes (COI). This paper shows the formal representation of the policy in the cloud computing environment.

### 1.2 Extended Definition

Chinese wall policy uses the concept of Conflict of Interest classes to implement security. The companies which are in competition with each other are placed in one group. This group is known as Conflict of Interest Class. If a company tries to access an object within the same CIR class then the access is denied.

The name itself explains everything. A Chinese wall is a kind of wall that is placed between competing companies to secure their confidential information from each other. For example, Figure1 illustrates the concept of Chinese Wall Security.

The above figure shows three people X, Y and Z. If X & Y are friends and Y & Z are friends and also X and Z are enemies, then it should not happen that confidential information of X goes to Z via Y. The cross on the line between Y and Z means that Y cannot pass the information of X to Z. For this reason a Chinese Wall is placed between X and Z.

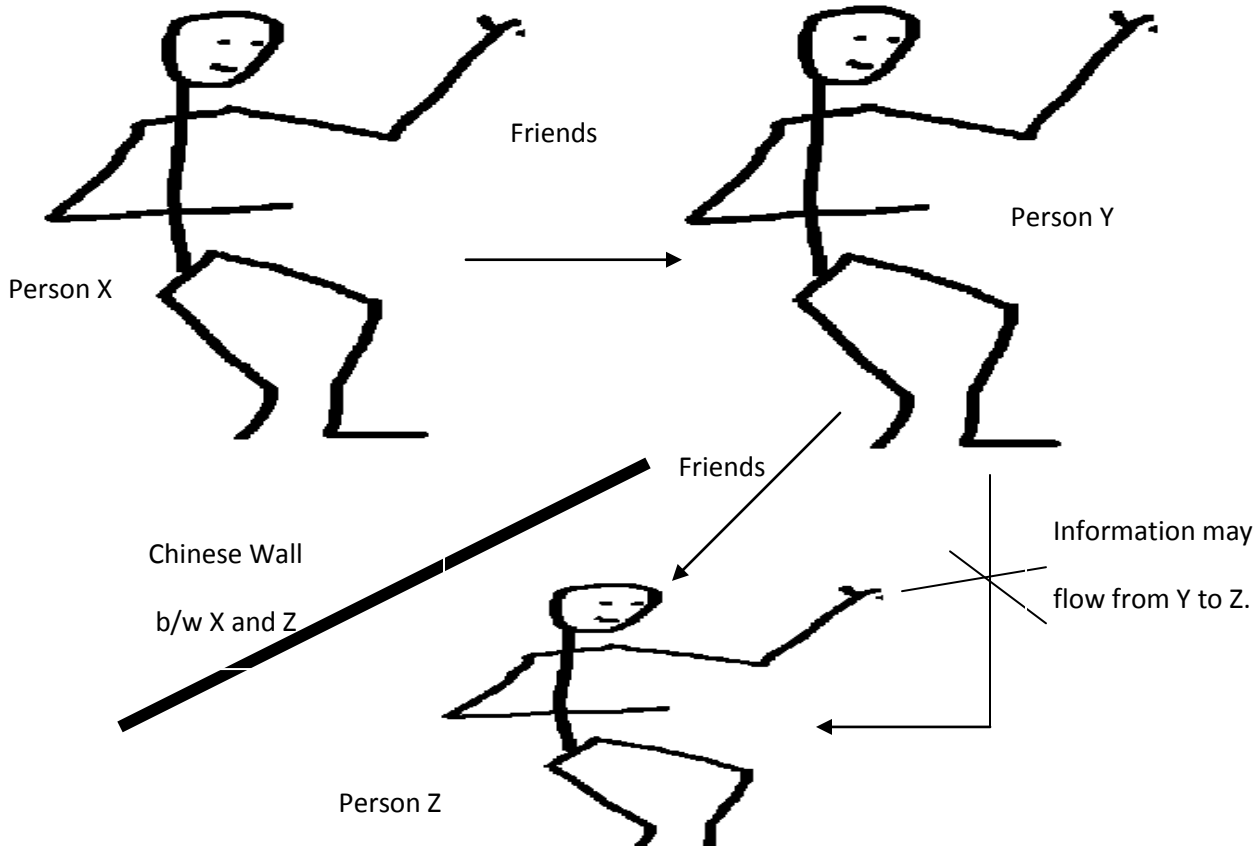



**Figure 1: Concept of Chinese Wall Security Policy**

Thus, the Chinese wall policy combines commercial discretion with legally enforceable mandatory controls [Bre89]. It is required in the operation of many financial services organizations and is, therefore, perhaps very significant to the financial world.

## 2. Why Chinese Wall Security Policy?

The previous chapter just gives an introduction to my project. This chapter will give a more detailed description of Cloud Computing and why do we need Chinese wall security policy.

### 2.1 Cloud Computing

First, came the traditional software. The user paid a one-time fee and owned the software and stores it on his computer. For the whole life time, this software now belongs to the user and he can use it any time.

Then a very nice service came known as software as a service (SaaS). It proved to be very useful to the business services and customer services. In this kind of service, the software is not stored on a user's computer instead it is accessed through internet. For example, the mail systems these days like Hotmail or Gmail. Now the user, don't have to worry about the storage requirements. That is totally the vendor's problem.

But this "one size fits all" approach didn't work out for very large companies. The companies require their data to be inside the firewall and moreover the cost vs. usage graph for them didn't prove to be beneficial to them.

The ideal thing that was best suited for them was to have some service which could provide them with following features:

- Convenience and simplicity of SaaS
- Flexibility of traditional software

In other words they needed a hybrid of SaaS and traditional software that allow any software to run as a service in a data center that is owned and managed by someone else. But the problem was that an application is so hard to deploy in a new environment.

All these problems lead to a new concept known as Virtualization. The infrastructure and the applications are not dependent one each other. They are totally independent. Various applications can be shared a server and an application can run on many servers.

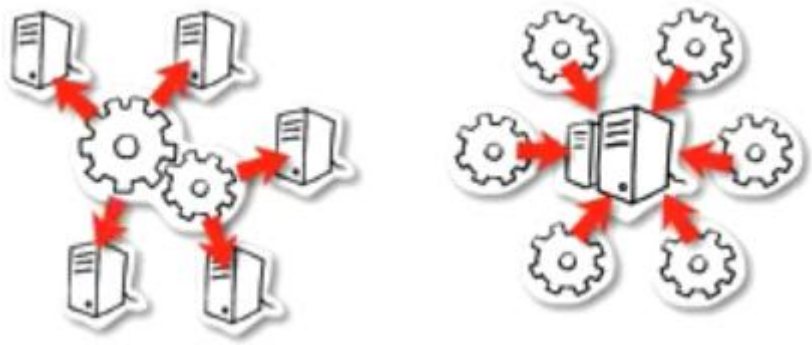


**Figure 2: Infrastructure and applications**

This is possible only if the application is virtualized. Virtualization means that everything is packed with the application i.e. the database, middleware and the operating system, all are packaged with the application. This package can run anywhere which means that it's not necessary that it runs on your data center or application provider's data center. Instead now it can run on a cloud. Cloud is a service that provides you computing resources as you need them. In other words it charges you for the services only when you use them. This "Pay as you go" service was the basic idea behind Cloud Computing.

## 2.2 Security Threats in Cloud Computing

Cloud Computing Services are growing day-by-day but the problem is that all the computation in cloud computing is done on a data center. Thus, it brings along with it a lot of security threats. When we are accessing the internet, the two biggest factors that ensure security are identity and access control. It has not been long to the concept of cloud computing for the customers to completely trust it. The problem with cloud computing is that your data now goes directly to the third party.

Confidentiality is a very big problem for data-at-rest, or stored data, since the IT professionals have to trust the security of the third-party storage. Also, the data-in-motion can be intercepted by someone. Basically, with cloud computing there is a free flow of information. So the users should have a choice to know that with which users his information is secured and with whom it's not.

So, Chinese wall security comes in picture over here. This policy allows users to secure their confidential information. It allows information to flow to only those users who at any point will not let the information flow to the enemies of the former. In other words, with the help of this policy, each user is forced to pass through the secure DAC and only if the system is secure after adding the user then it allows the flow of information. Thus the users can be relieved about the security of their confidential information.

## 3. Background

Now that we know, why exactly we need Chinese Wall Policy, it is important to understand the main concepts on which this policy is based. This policy is based on the concept of Conflict of Interest (COI). To understand the concept of COI, you should very well understand the concept of equivalence relation. This chapter gives a basic understanding of the important concepts in Chinese Wall Security Policy.

### 3.1 Equivalence Relations

Before going into the detail of what an equivalence relation is, it is necessary to know what a relation is. Suppose there is a set Z. The subset of Cartesian product of Z with itself is a relation R on Z. Hence any particular element, x, of Z has the relation R with any element, y, of Z if and only if (x, y) is an element of R.

To understand the concept of equivalence relations, it is necessary to know about reflexive, transitive & symmetric relation and also equivalence classes and partition.

A relation on a set say, B is reflexive if bRb for all b$\in$ B. A relation on a set say, B is transitive if for all (a; b; c) $\in$ B, if aRb and bRc then aRc. A relation on a set say, B is symmetric for (a; b) $\in$ B if aRb then bRa also holds true. Therefore a relation is an equivalence relation if it is Reflexive, Transitive and Symmetric.

### 3.2 Conflict of Interest

Conflict of Interest is the heart of Chinese wall. If there is someone who has his/her own interests professionally and also he/she is in the position of trust then that condition is

said to be Conflict of Interest. Such competing interests can make it difficult to fulfill his or her duties impartially. A conflict of interest exists even if no unethical or improper act results from it. A conflict of interest can create an appearance of impropriety that can undermine confidence in the person, profession, or court system [May08]. A conflict can be mitigated by third party verification or third party evaluation noted below—but it still exists. In terms of financial world, to protect the private information of companies from each other, the companies which are not in competition with each other are placed in one group. This group is known as Conflict of Interest (COI) class. So information can be passed only within the same conflict of interest class.

In the Section 7.1 in [Matt03], it is assumed that a set of objects could be partitioned into mutually disjoint CIR-classes (conflict of interest classes). Note that a partition induces an equivalence relation and vice versa which has been stated above. Hence BN's assumption implies CIR is an equivalence relation. In the following example, we shall show that CIR is not. Let O = {USA;UK; USSR} where O is set of objects and CIR be the conflict of interest binary relation among three countries. CIR can be read as "in cold war with." If CIR were transitive, then the following two statements:

XYZ is in a cold war with ABC, and

ABC is in a cold war with DEF,

would imply that XYZ is in cold war with DEF.

Obviously, the last statement is absurd. This example can easily be generalized: Assume X and Y are "friend", and both have conflict of interests with Z. That is X and Z are in

the same CIR-class. By the same reason Y and Z are in the same CIR-class. On the other hands, since X and Y are "friend"; they should not be in the same CIR-class (by definition). These two CIR-classes are distinct but overlapped.

So we conclude that CIR cannot be an equivalence relation (reflexive, symmetric and transitive binary relation). If CIR is not an equivalence relation, CIR-classes do not form a partition. In other words, CIR-classes do overlap [Lin03]. In BN's (Brewer and Nash) language, company data that are in conflict can be in the same side of Chinese wall. Hence, BN's theory collapses. Thus, Conflict of interest is not an equivalence relation. The example given in section 7.1.1 in [8] is not COI class. It is a same business so it is an equivalence relation. So this is not a proper example of COI.

Thus, COI as a binary relation O X O has the following properties:

COI 1: Symmetric

COI 2: Anti-Reflexive

COI 3: Anti-Transitive.

It is very clear that COI-2 is mandatory as a company cannot conflict to itself. If a company X is a enemy of some company Y then Y is definitely a enemy of X. Thus COI-1 is valid.

## 4. Chinese Wall Security Model

The whole idea behind Chinese Wall was not to let secret information of a company to be revealed to the competitor companies through the financial consultants.

All corporate information is stored in a hierarchically arranged filing system as shown in Figure 3 [Ber]. There are three levels of significance [Bre89]: at the bottom level, individual pieces of information are considered, each representing a single corporation. This information is stored in filed called as objects; at the medium level, all the objects

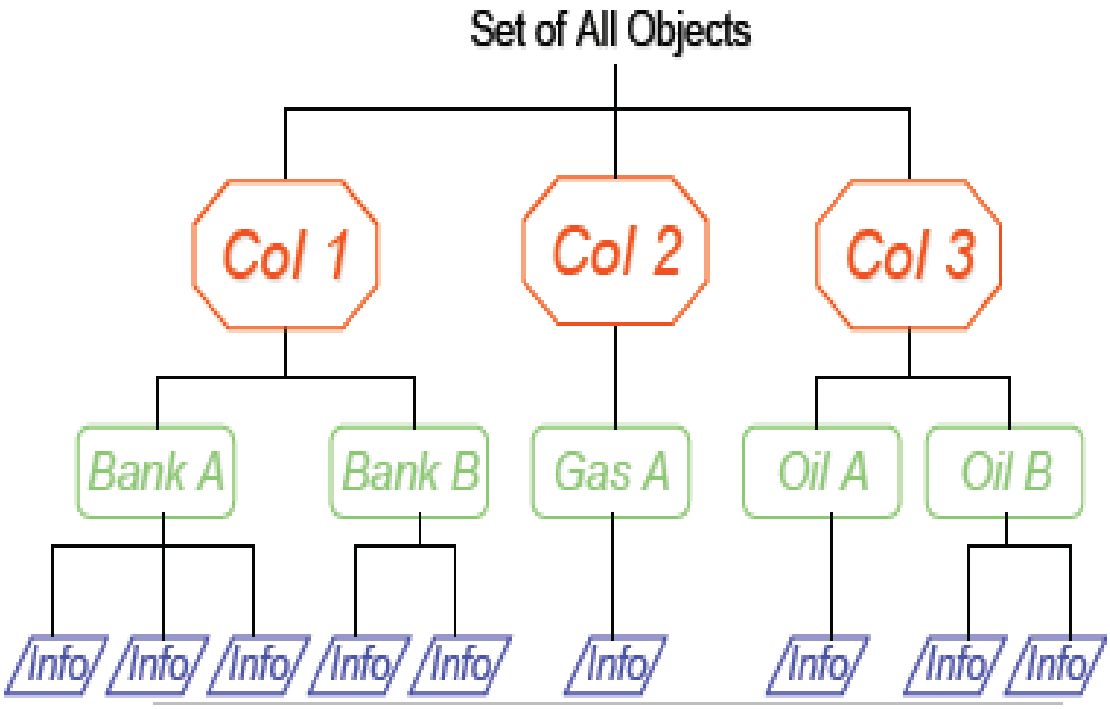



**Figure 3: Data Classification [Ber]**

from the same corporation are grouped into one company dataset; at the top level, all these company datasets from competing corporations are grouped together. This group is known as conflict of interest class.

There are two things that are always associated with the name of the object:

- Company dataset
- Conflict of Interest class

The basis of the Chinese Wall policy is that people are only allowed access to information which is not held to conflict with any other information that they already possess [Bre89]. The only information that a user can have is the one that he/she already had on their computer and the one that they had already accessed. Thus, in consideration of the Bank-A, Bank-B, Gas Company-A, Oil Company-A and Oil Company-B datasets mentioned previously, a new user may freely choose to access whatever datasets he likes; concerning the computer a new user cannot have any conflicts since they does not possess any information. Sometime later, however, such a conflict may exist.

Suppose the user requests to get the data for Oil Company-A. So it will get the access to the data since he/she is a new user and thus no conflict exists. Now if after sometime the same user asks for the data of Bank-A then he/she will be granted the access to the data since they belong to different conflict of interest classes. Up till this point everything is fine since there is no conflict. Now if the user requests for accessing the data of Oil Company-B then it the request will be denied since they belong to the same conflict of interest class.

A new user has complete freedom to access anything he wants to choose. After the user makes the initial choice, a Chinese wall is built around the dataset for that user and the opposite or the wrong side of this wall can be considered as any dataset in the same conflict of interest class [Bre89]. The user always have access to the dataset in the different conflict of interest class but whenever he/she accesses some new data from a different COI then the Wall around him/her changes to include that dataset. So it can be said that combination of mandatory control and free choice is Chinese wall.

## 5. Literature Review

To provide some new ideas and well understanding of methods that can be applied to solve this problem, I have reviewed some literatures, on the following areas which are described below.

### 5.1 Rules for Chinese Wall Security policy

One of the papers [Ber] explained the access rules which helped me a lot in implementing this project:

**Read Rule**: This rule states that an object O can be read by a subject S if and only if the object is in the dataset that was previously accessed by A and also the object O is in the CoI from which S had not accessed anything [Ber].

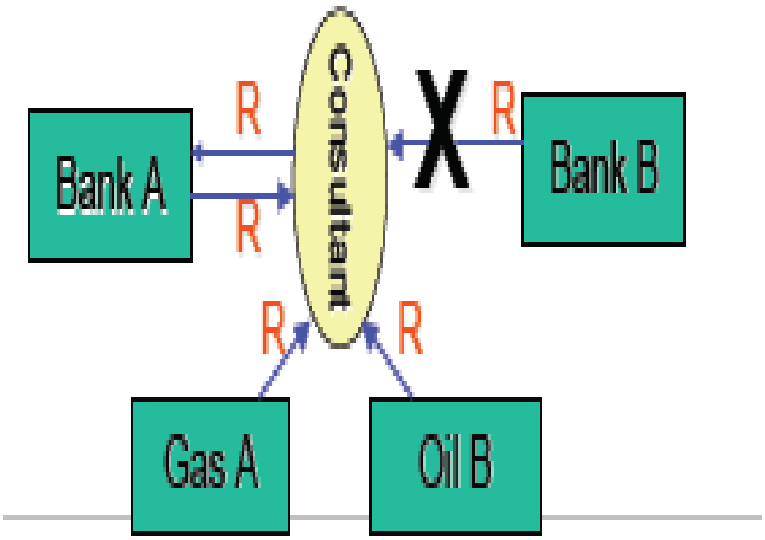



**Figure 4: Read Rule [Ber]**

**Write Rule:** A subject S can write an object O if S can read O according to the Read Rules and no object has been read by S which is in a different company dataset to the one on which write is performed [Ber].

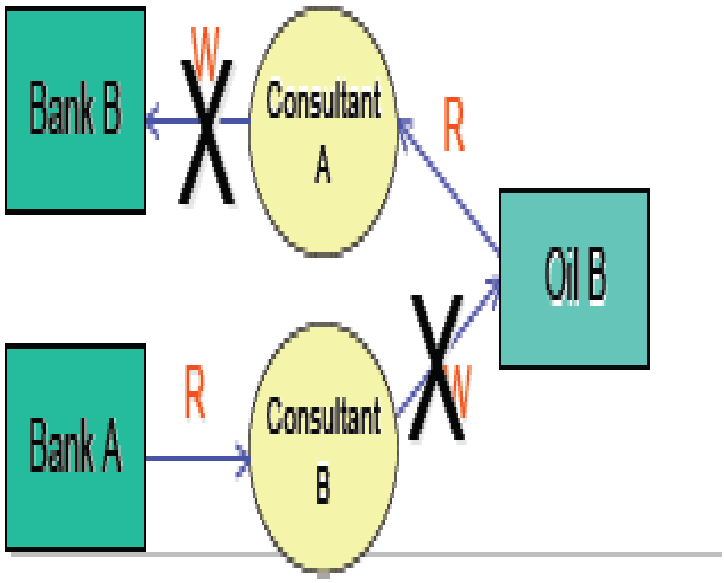



**Figure 5: Write Rule [Ber]**

### 5.2 Alternative method for providing security in financial world

In the paper [Bre89] an alternative method known as Bell LaPadula model was explained which provides security in financial world. It helped me a lot in understanding how security is enforced.

The BLP model places no constraints upon the interrelationships between objects, in particular it does not require them to be hierarchically arranged into company datasets and conflict of interest classes. Instead it imposes a structure upon the security attributes themselves. Unlike the Chinese Wall policy, BLP attaches security attributes to subjects

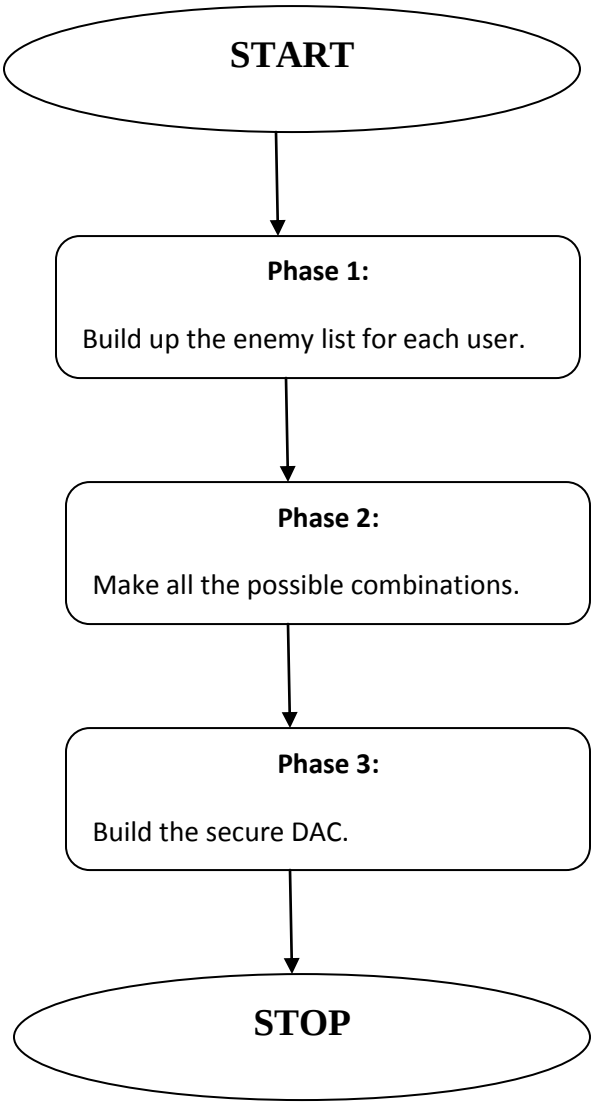
as well. They are complementary to object labels and have the form (*clear*, *NTK*) where *clear* is the subject's clearance, i.e. the maximum classification to which he is permitted access and *NTK* is the subject's need-to-know, i.e. the sum total of all categories to which he is permitted access. BLP only works if subjects are not given the freedom to choose which company datasets they wish to access. In other words, these transformations totally ignore the free choice nature of the Chinese Wall policy. This freedom of choice can be restored (e.g. by extending subject need-to-know to cover all company datasets) but only at the expense of failing to express the mandatory controls. The Chinese Wall model is therefore distinct from BLP and important in its own right.

## 6. Idea behind the Implementation

So far I have given the theoretical explanation of my whole project. Now, this chapter explains the actual method used for implementing this project.

### 6.1 Flowchart

The implementation of this project has been divided into three phases as shown below in the flowchart.



## 6.2 Methodology

As shown above this project includes three phases:

**a) Phase 1:** Build up the enemy list for each user.

Suppose there are 3 companies A, B and C. A and B are friends, B and C are friends, and A and C are enemies. Now A gives his information to B. Since B and C are friends so B can give all the information to C. But if this happens then C who is the enemy of A gets to see the information of A. This should not happen.

Now the possible combinations for enemy list of user ‘a’ are as follows:

a -> -

a-> b

a-> c

a-> b, c

The first case states that user ‘a’ has no enemy.

The second case states that ‘b’ is an enemy of a. But remember one thing. This is not reflexive. That means that a-> b states that ‘b’ cannot read the data of ‘a’. But it is possible that ‘a’ can read the data of ‘b’.

The third case states that ‘c’ is an enemy of ‘a’.

The fourth case states that ‘b’ and ‘c’, both cannot read the data of ‘a’ or it means that b and c are enemy of ‘a’.

In a similar way the possible combinations for enemy list of user ‘b’ and user ‘c’ are as follows:

**Combinations of ‘b’:**

b -> -

b-> a

b-> c

b-> a, c

**Combinations of ‘c’:**

c-> -

c-> a

c-> b

c-> a,b

So each user has $2^{n-1}$ combinations for enemy list.

The function used for implementing this functionality is as follows:

```
int fillFirstRound ( int nUsers, vector<char> users )
{
   int nElements;
   int nCombination;
while (in) {
   //cout << str << endl;
   oneDvector.push_back(str);
   getline(in,str);
   if ( str == "-" ) {
      twoDvector.push_back(oneDvector);
```

```
        oneDvector.empty();
        oneDvector.clear();
    }
}

combinations_recursive(twoDvector, twoDvector.size(), pos, 0, 0 );
    vector <char> combination;
    typedef vector <char> oneDVector;
    vector < oneDVector > levelOneCombination;
    for (int i = (nUsers-1) ; i >= 0 ; i-- ) {
        int nElements = nUsers-1 ;
        int nCombination = calculate( nElements, i);
        vector <char> elems;
        //cout << "for " << i << " elements " << nCombination << " combinations" << endl;
        for(int copyIndex = 1; copyIndex<users.size(); copyIndex++)
            elems.push_back(users[copyIndex]);
        vector<unsigned long> pos(i);
        combinations_recursive(elems, i, pos, 0, 0);

for (int k = 0; k < nCombination ; k++ ) {
            for(int j = 0; j < i; j++) {
                vector<char>temp(i);
```

```
            //creating combinations
            int base = j;
            temp.push_back
            combination.push_back(users[base+1]);
            if( j+1 == users.size() ){
               endReached = true;
            }
         }
         levelOneCombination.push_back(combination);
         combination.clear();
      }
   }
   for(int i=0; i < levelOneCombination.size() ; i++) {
      for(int j=0; j < levelOneCombination[i].size() ; j++) {
         cout << levelOneCombination[i][j] << " ";
      }
      cout << endl;
      cout << "Row no " << i << "finished " << endl;
   }
}
```

**b) Phase 2:** Make all the possible combinations.

Now, the system will output all the possible combinations of these enemy lists. For example one of the combinations is:

a->-

b->c

c->a, b

That means picking one row from each user's enemy list and combining them.

This one combination tells about the whole system. In other words it states that which user or company can read which company's data.

So finally, if there are three users, then the output of the system will be as follows:

| Combination 1 | Combination 2 | Combination 3 | Combination 4 |
|---|---|---|---|
| a->- | a->- | a->- | a->- |
| b->- | b->- | b->- | b-> - |
| c->- | c->a | c->b | c-> a, b |

Similarly, there will be other combinations. Thus for three users, there will be 64 combinations. So, for n users the total possible combinations will be $(2^{n-1})^n$.

The function used for implementing this functionality is as follows:

```
void combinations_recursive (const vector<char> &elems, unsigned long req_len, vector<unsigned long> &pos, unsigned long depth, unsigned long margin)
{
  // Have we selected the requested number of elements?
  if (depth >= req_len) {
```

```
  static int comb = 1;
  cout << comb++ << ") " ;
  for (unsigned long ii = 0; ii < pos.size(); ++ii)
    cout << elems[pos[ii]];
  cout << endl ;
  return;
 }

 // Are there enough remaining elements to be selected?
 // This test isn't required for the function to be
 // correct, but it saves futile calls.
 if ((elems.size() - margin) < (req_len - depth))
  return;

 // Try to select new elements to the right of the last
 // selected one.
 for (unsigned long ii = margin; ii < elems.size(); ++ii) {
  pos[depth] = ii;
  combinations_recursive(elems, req_len, pos, depth + 1, ii + 1);
 }
 return;
}
```

**c) Phase 3:** Build the secure DAC.

Now the final step is to provide the users with the secure combinations. Out of the $[2^{(n-1)}]^n$ there are only few combinations that are secure. Secure combination means the combinations which will not violate Chinese wall. In other words, there will be no such situation in which the information of a competing company will go to another company via some friends.

Now, for building up the secure combinations or a secure DAC I have used the method of trajectory.

Now, suppose one of the combinations in a system of 5 users is:

E (a) = b, c

E (b) = c, d

E (c) = d, e

E (d) = a, b

E (e) = b, c

According to the above system, User a has two enemies: b and c. This means that b and c cannot read User a's data. It will be similar for the users. So the above combinations is called enemy list. Thus E(a) represents enemies of a.

Now if we take the compliment of the above system then that will represent the friend list. In other words, the friend list will be the exact compliment of the enemy list. So it can be said that the friend list is an equivalence relation. It will look like the following:

F (a) = a, d, e

F (b) = a, b, e

F (c) = a, b, c

F (d) = d, c, e

F (e) = a, d, e

Here, F (a) represents the users who are allowed to access or see the data of a. In other words they are the friends of a.

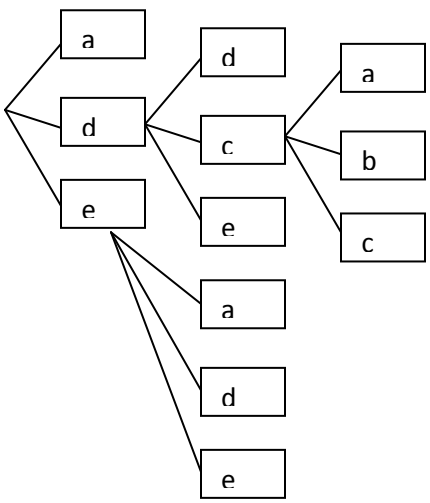
Now, from the friend list we can build up the trajectory. A trajectory is a path of all users who can in any way access the data of the other user. It will be clearer with an example. Consider the above system only. We will build the trajectory using the friend list.

Now T (*) represents the trajectory of a single user. In other words, which ever users comes in the trajectory of the user, whether or not they are the friends of that user, will be the users who are able to access the data of the corresponding user. So a combination is only secure if and only if the intersection of the trajectory of the user and the enemy list of the corresponding user is null. In other words,

T (*) ∩ E (*) = Ø

So the trajectory of user ‘a’ will be:

T(a) = a



Thus, the trajectory of ‘a’ is:

T (a) = {a, d, e, b, c}.

This means that every user gets to access the data of ‘a’ in one or another way, though they are not allowed to access it. This means that the combination mentioned above is not secure and thus violates Chinese wall.

This secure system is known as Chinese Wall Discretion Access Control System and thus a secure system which obeys Chinese wall has following properties:

a) Anti-transitive

b) Anti-reflexive

c) Symmetric

d) T (*) ∩ E (*) = ∅

Or in other words it can be said that in such a system the friend list will always be an equivalence relation.

## 7. Real world application of the Chinese Wall Policy

This chapter explains the actual application of Chinese wall policy by applying it to a real- world cloud computing model.

### 7.1 History of Hadoop

Hadoop was created by Doug Cutting, the creator of Apache Lucene, the widely used text search library. Hadoop has its origins in Apache Nutch, an open source web search engine, which is itself a part of the Lucene project.

In January 2008, Hadoop was made its own top-level project at Apache, confirming its success and its diverse, active community.

### 7.2 What is the Hadoop Cloud solution?

Hadoop is an incarnation of Map/Reduce in a Cloud environment. Map/Reduce is for huge data sets that have to be indexed, categorized, sorted, culled, analyzed, etc. It can take a very long time to look through each record or file in a serial environment. Map/Reduce allows data to be distributed across a large cluster, and can distribute out tasks across the data set to work on pieces of it independently, and in parallel. This allows big data to be processed in relatively little time.

It takes a large data set and then breaks it down into smaller data sets. Thus it has some potential uses:

- indexing large data sets in a database
- image recognition in large images
- processing geographic information system (GIS) data -

- analyzing unstructured data

So basically it can be used in any situation where processing a data set would be impractical due to its size.

**7.3 How it works?**

Hadoop is an example of cloud computing framework which was developed by Doug Cutting in 2006. There are several subprojects that are provided by Hadoop:

- Avro
- Chukwa
- HBase
- HDFS
- Hive
- MapReduce
- Pig

These all subprojects provide several capabilities like distributed processing of large data, structured data storage, high throughput access to application data etc. Because of these capabilities, Hadoop is being used by large Content-Distribution companies, such as:

- Yahoo!
- A9
- New York Times
- Facebook
- Veoh

Hadoop has a very simple programming model:

- Map (anything) -> key, value
- Sort, partition on key
- Reduce (key, value) -> key, value

There is no parallel processing or message passing semantics. It is programmable in Java or any other language (streaming). It creates or allocates a cluster and then puts data onto the file system; Data is split into blocks and stored in triplicate across the cluster.

Then, the job can run. The Map code is copied to the allocated nodes, preferring nodes that contain copies of the data. It monitors workers, automatically restarting failed or slow tasks. It gathers output of Map, sort and partition on key. It runs reduce tasks. The results of the job are now available on the Hadoop file system.

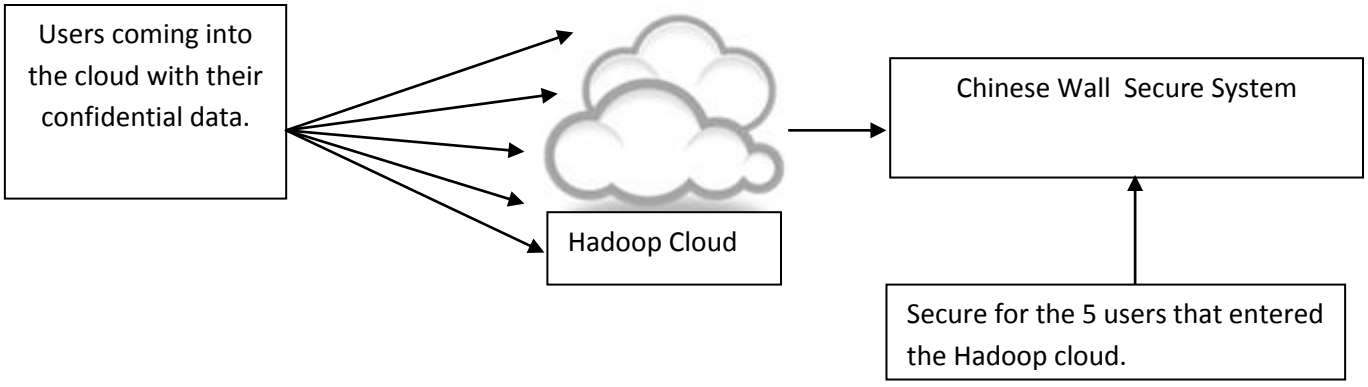
### 7.4 Security Issues with Hadoop

Hadoop is increasingly useful but here are some security issues with it. Hadoop holds data in HDFS - Hadoop Distributed File System. The file system has no read control; all jobs are run as 'hadoop' user, and the file system doesn't follow access control lists. The client identifies the user who is running a job by the output of the 'whoami' command - which can be forged. Thus, there is no read or write control. Any business running a Hadoop cluster gives all programmers and users the same level of trust to all the data that goes into the cluster. Any job running on a Hadoop cluster can access any data on that cluster.

The kind of security concerns may be resolved by applying the Chinese Wall Policy to Hadoop Cloud. So Hadoop basically needs a solution that sits on the file system. The

problem is that the access control is held at the client level. The solution is that it should be at the file system level. Access control list checks should be performed at the start of any read or write. Basically Hadoop can use the secure DAC (Discretionary Access Control) that support the Chinese Wall Policy. The secure DAC will always satisfy the three properties of an equivalence relation; Symmetry, Reflexivity, Transitivity. These properties are the base of Chinese Wall policy. So if the users are passed through this secure DAC then it will make sure that the whole system is secure.

So, now if there are 5 users who enter into the Hadoop Cloud, then they are internally passed through the secure system to check that their sharing requirements meet Chinese Wall security policy.




Now, after sometime if a new user enters into the system then the Hadoop cloud will pass the 6 users through the secure system again. The secure system has been implemented incrementally so whenever a new user comes into the system it will not run the algorithm for all the users again. Instead the system will already have secure system for 5 users, so it will just add the $6^{th}$ user to the DAC and check if the $(n+1)^{th}$ user can be safely integrated into the system.

This is one of the real world applications where Chinese Wall Policy may be applied. There are many other applications that can use this policy or model for providing security to the users.

## 8. Conclusion & Future Work

In this paper I have explored the security threats with cloud computing and how the security can be enforced for the users on the data center. With the help of a formal representation and a small implementation of the Chinese Wall Model I have tried to secure the information of competing companies from each other. If someone doesn't obey this policy then it is considered as not professional and fraudulent. Thus this is a real commercial policy which can be formally modeled. The implementation has been done in C++ language which works great and very fast. With the completion of the implementation, the results were able to prove the approach and its reasoning done so far. As of now the limitation of this approach can be that it won't be able to recognize the database users. So this can be taken care of in the future implications on the same issue. Studies and researchers have predicted the bright future of Chinese Wall Security, as they say that it is applicable to solve any simple to complex issues in the financial world.

**Works Cited**

[Ber] Bertino, E. (1998, May Thursday). *Purdue University.* Retrieved November Tuesday, 2008, from homes.cerias.purdue.edu/~bhargav/cs526/security-4.ppt

[Bre89] Brewer, D. F., & Nash, M. J. (1989). THE CHINESE WALL SECURITY POLICY. *THE IEEE SYMPOSIUM ON RESEARCH IN SECURITY AND PRIVACY*, (pp. 206-14). Oakland.

[Lin03] Lin, T. (2003). Chinese Wall Security Policy Models:Information Flows and Cinfining Trojan Horses. *Data and Applications Security XVII: Status and Prospect*, (pp. 275-287). Estes Park.

**Bibliography**

[Ber] Bertino, E. (1998, May Thursday). *Purdue University.* Retrieved November Tuesday, 2008, from homes.cerias.purdue.edu/~bhargav/cs526/security-4.ppt

[Bre89] Brewer, D. F., & Nash, M. J. (1989). THE CHINESE WALL SECURITY POLICY. *THE IEEE SYMPOSIUM ON RESEARCH IN SECURITY AND PRIVACY*, (pp. 206-14). Oakland.

[Dem95] Demurjian, S. (1995). The Multimodel and Multilingual Database Systems-A Paradigm for the Studying of Database Systems. *IEEE Transaction on Software Engineering*, (pp. 10-19). San Fransisco.

[Dem97] Demurjian, S., & Ting, T. (1997). Towards a Definitive Paradigm for Security in Object-Oriented Systems and Applications. *J.of Computer Security*, (pp. 352-367). Oxford.

[Hs70] Hsiao, D., & Harary, F. (1970). A Formal System for Information Retrieval From Files. *Communications of the ACM*, (pp. 24-29). Corrigenda.

[Lin2000] Lin, T.Y (2000). Chinese Wall Security Model and Conflict Analysis. *The 24th IEEE Computer Society International Computer Software and Applications Conference*, (pp. 246-269). Taipei.

[Lin03] Lin, T.Y (2003). Chinese Wall Security Policy Models:Information Flows and Cinfining Trojan Horses. *Data and Applications Security XVII: Status and Prospect*, (pp. 275-287). Estes Park.

[Lin 89] Lin, T.Y (1989). Chinese Wall Security Policy--An Aggressive Model. *Proceedings of the Fifth Aerospace Computer Security Application Conference*, (pp. 286-293). Taipie.

[Lin02] Lin, T.Y (2002). Placing the Chinese Walls on the Boundary of Conflicts - Analysis of Symmetric Binary Relations. *Proceedings of the International Conference on Computer Software and Applications*, (pp. 966-971). Oxford.

[Matt03] Matt, B. (2003). *COMPUTER SECURITY: Art and Science.* San Jose.

[Paw] Pawlak, Z. (1997). Analysis of Conflicts. *Joint Conference of Information Science*, (pp. 350-352). North Carolina.

[Ter88] Teresa, L. F. (1988). Access Control Polices for Database Systems . *The 1988 Workshop on Database Security,*, (pp. 34-39). Ontario.

[May08] May, T. A. (2008). Retrieved 2009, from Conflict of Interest- Michigan State University: https://www.msu.edu/~bi

[Ting88] Ting, T. (1988). A User-Role Based Data Security Approach. *Database Security: Status and Prospects* , North-Holland, 1988.