Mobile Agent-Based Authentication: A Model for User Authentication in a Distributed System

Dilli Prasad Sharma
Department of Computer Science and Information Technology
Prime College, Kathmandu, Nepal

ABSTRACT
Security issues in a distributed system are always crucial and existing distributed computing security technologies do not adequately address for its scalability, performance and heterogeneity. In this paper, an agent-based authentication model is designed and uses a mobile agent which is an object that migrates through many nodes of a heterogeneous network of computers, under its own control in order to perform designated tasks using local resources of the nodes. A mobile agent is called authentication mobile agent that can migrate from machine to machine in the agent-enabled network and responsible for providing the authentication service in the distributed system. The authentication mobile agent uses digital signature algorithm for the authentication of the mobile code, and password encrypted with a secret key for the user authentication. The agent maintains a small database file that also migrated with it. It may again move towards the other machines on demand. This approach provides many benefits to the development of distributed applications.

General Terms
Security in a Distributed System, Digital Signature Standard

Keywords
Distributed Systems, Scalability, User Authentication, Digital signature, Mobile Agents, Mobile Code Migration, Digital Signature Algorithm (DSA)

1. INTRODUCTION
A distributed system [1] is a collection of independent computers that appears to its users as a single coherent system. The transparency, openness, and scalable are the important characteristics of a distributed system. A distributed system should have transparent. The access, location, migration, replication, concurrency are the transparencies of a distributed system.

A distributed system is susceptible to a variety of security threats mounted by intruders as well as legitimate user of the system. The security concern in distributed system can be divided into two major parts [1]. First part concerns secured communication between users and processes, possibly residing on different machines of heterogeneous systems. The principle mechanism for ensuring secure communication is that of the establishment of a secure channel. The secure channel should be able to provide the security mechanisms including authentication, integrity, and confidentiality. The use of secured socket layer and transport layer security enables the security concern for secured channel. Second part concern authorization which deals with ensuring that a process gets only those access rights to the resources in a distributed system it is entitled to and a common mechanism to ensure it is access control.

An authentication mechanism is used to verify the claimed identity of a user, client, and server and so on. Authentication can be of one way or mutual. The authentication In the case of clients, the basic premise is that before a service will do work for a client, the service must learn the client's identity. After a client has been authenticated, it is necessary to check whether that client is authorized to perform the action requested that depends on who accesses the data and what permission granted to read, to modify, to add or to remove a record in a database. The uses of auditing tools are to trace which clients accessed what data or resources and which way.

In this paper, it is expressed the study of the research literatures on mobile agents, distributed systems, and security mechanisms in distributed system and building architecture of mobile agent that provides certain kind of security services like as a user authentication service in a distributed system. The main contribution of this research is to find out the applicability of mobile agents in distributed system and proposed an agent called authentication mobile agent (AMA) which is responsible for providing the authentication services in a distributed system. The AMA migrates from one machine to another machine of the agent-enabled network. When the authentication service is required in a particular machine, the machine multicast a request signal for AMA services, and the AMA come over there and handle services. The AMA uses the password-based authentication. The AMA receives the password and verifies it and then provides the authentication services. For the of the authentication services, it maintains a small database or code that also migrated with it. The AMA may again moves towards the other machines on demand.

2. MOBILE AGENT PARADIGM
A mobile agent (MA) is a program that can migrate from machine to machine possibly in a heterogeneous network. The program chooses when and where to migrate. It can suspend its execution at an arbitrary point, transport to another machine and resume execution on the new machine. Mobile agents can migrate from host to host in a network, at times and to places of their own choosing. The state of the running program is saved, transported to the new host, and restored, allowing the program to continue where it left off. Mobile-agent systems [6] differ from process-migration systems in that the agents move when they choose, typically through a ``jump" or ``go" statement, whereas in a process-migration system the system decides when and where to move the running process.

Mobile agents [11] are an effective choice for many applications, for several reasons, including improvements in latency and bandwidth of client-server applications and reducing vulnerability to network disconnection. Although not all applications will need mobile agents, many other applications will find mobile agents the most effective implementation technique for all or part of their tasks.

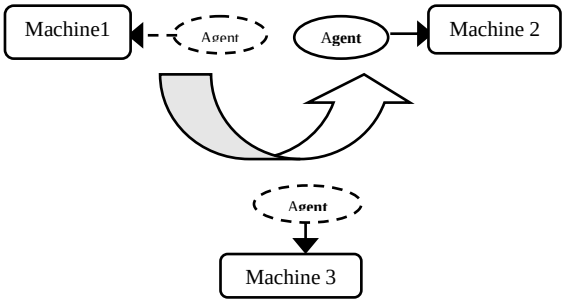

Figure 1: A mobile agent paradigm

3. LITERATURE REVIEW

Initially to simplify the communication in a distributed system the concept of the code migration was proposed. The code migration technique simplifies the task of distributed system by executing the code locally. Traditionally, code migration in distributed systems took place in the form of process migration in which an entire process was moved from one machine to another. The models for the code migration are either weak mobility or strong mobility.

The code migration framework consists of three segments [8]. The code segment is the part that contains the set of instructions that make up the program that is being executed. The resource segment contains references to external resources needed by the process, such as files, printers, devices, other processes, and so on. Finally, an execution segment is used to store the current execution state of a process, consisting of private data, the stack, and the program counter. The bare minimum for code migration is to provide only weak mobility. In this model, it is possible to transfer only the code segment, along with perhaps some initialization data, for example, with Java applets.

In contrast to weak mobility, in systems that support strong mobility the execution segment can be transferred as well. The characteristic feature of strong mobility is that a running process can be stopped, subsequently moved to another machine, and then resume execution where it left off. Clearly, strong mobility is much more powerful than weak mobility, but also much harder to implement..

There is a strong case for the use of mobile agents in many Internet applications [6] and, there is a clear evolutionary path that will take us from current technology to widespread use of mobile code and agents within the next few years. Once several technical challenges have been met, and a few pioneering sites install mobile-agent technology, use of mobile agents will expand rapidly.

Mobile agent and client/server performance [5] with some experimental data shows the mobile agents lead to faster applications, reduced bandwidth demands, or less dependence on a reliable network connection. The author in the paper [2] explained the applications of mobile agents in electronic commerce, network management, personal digital assistant. The advantages of using mobile code and mobile agent computing paradigms have been proposed [3].These advantages include: overcoming network latency, reducing network load, executing asynchronously and autonomously, adapting dynamically, operating in heterogeneous environments, and having robust and fault tolerant behavior.

There are a several mobile agent systems [4] are available. Some of them are: Agent TCL, ARA, Concordia, Mole, Tacoma, Aglets, and Voyager. Though these systems differ in their goals, motivations, and implementations, they all provide common functionalities that support: the migration of agents, the communication between agents, various programming/interpreted languages, and various forms of security.

4. CRYPTOGRAPHIC ALGORITHMS AND AUTHENTICATION PROTOCOLS

The cryptographic algorithms [7] are the fundamental to security in distributed systems. There are fundamentally two categorizes of encryption algorithms symmetric and asymmetric. Symmetric encryption is forms of cryptosystem in which encryption and description are perform using the same key. It is also called conventional encryption or single key or private key encryption. In symmetric encryption, the sender and recipient share a common key.The symmetric encryption transform plaintext into cipher text using a secret key and an encryption algorithm. The cryptographic algorithms such as DES, TDES, AES, IDEA, Blowfish, RC5, and CAST-128 are the symmetric encryption algorithms.

In 1978, Needhm and Schroeder published a authentication protocols are the heart of many security techniques used in the distributed systems. Authentication Protocols are used to convince parties of each other's identity and to exchange session keys. They may be one-way or mutual. Central to the problem of authenticated key exchange are two issues: confidentiality and timeliness. To prevent masquerade and to prevent compromise of session keys, essential identification and session key information must be communicated in encrypted form. This requires the prior existence of secret or public keys that can be used for this purpose. The second issue, timeliness, is important because of the threat of message replays.

4.1 Kerberos

The Kerberos Authentication Service was developed by the Massachusetts Institute of Technology to protect the emerging network services provided by Project Athena. The security of Kerberos depends on limited session lifetimes that is the period of validity of ticket granting server is generated limited to a few hours, the period must be long enough to avoid the incontinent interruption of service but short enough to ensure that users who have been de-registered or downgraded do not continue to use the resources for more than a short period. This might causes difficulties in some commercial environment.

4.2 X.509

The X.509 is an authentication protocol based on the use of public-key cryptography and digital signature. It defines a framework for the provision of authentication services by the X.500 directory to its users. It is recommended the uses the public key algorithm such as RSA and the digital signature is assumed to require the use of hash function. The heart of the X.509 scheme is the public-key certificate associated with each user.

4.3 Secure Socket Layer

The SSL protocol was originally developed by the Netscape Corporation. The SSL consists of two layers, an SSL record protocol layer, and handshake layer. The SSL record protocol layer implements a secured channel, encrypting and authenticating message transmitted through the connection-oriented protocol. The handshake layer consists the SSL handshake protocol. The SSL widely used to add a secured communication layer below existing application-level protocols. It is the de facto standard for use in application

requiring secured channel and available implementation in CORBA and Java.

4.4 Digital Signature Standard and Digital Signature Algorithm

Encryption, message authentication and digital signatures are all tools of modern cryptography. A signature is a technique for non-repudiation based on the public key cryptography. A digital signature is an authentication mechanism that enables the creator of a message can attach a code that acts as a signature. The signature is formed by taking the hash of the message and encrypting the message with the creator's private key. The signature guarantees the source and integrity of the message.

Digital Signature Standard (DSS) is the US Government approved signature scheme, which is designed to provide strong signatures without allowing easy use for encryption. The DSS makes use of the Secure Hash Algorithm (SHA), and presents a new digital signature technique, the Digital Signature Algorithm (DSA). The DSS was originally proposed in 1991 and revised in 1993 in response to public feedback concerning the security of the scheme. There was a further minor revision in 1996. Unlike RSA, it cannot used for encryption or key exchange. Nevertheless, it is a public key technique. DSA signatures use the message hash, global public values, private key & random k to create two part signature (s,r). This is verified by computing a function of the message hash, public key, r and s, and comparing the result with r.

5. MOBILE AGENT-BASED MODEL AND ITS IMPLEMENTATION

An authentication mobile agent (AMA) is an agent that can migrate from machine to machine in the agent-enabled network to provide the authentication services. The authentication service refers to the task of verifying the identity of the persons, or users, or the software agents connecting to a database. The security in distributed system can roughly be divided into two parts: first part concerns the secure communication. The principal mechanism for ensuring secure communication is that of secured channel. Secure channel specifically, authentication, message integrity, and confidentiality. The second part concern authorization, which deals with ensuring that a process gets only those. Interception, Interruption, Modification, and Fabrication are the security threats. The system should be able to protect itself against all possible security threats. The security policy describes the security requirements. A security policy describes precisely which actions the entities in a system are allowed to take and which ones are prohibited. The authors in paper [9] describes the design and implementation of policy engine for enforcing security policies in distributed object applications and integrated the their design as a part of Globe[10] system by identifying the number of security policy requirements in the context of replicated applications, expressed trust for different replica, and bridging the gap between an abstract security policy description and actual service implementation. The entities include users, data, services, machines and so on. The security mechanisms by which a policy can be enforced. Important security mechanisms are: encryption, authentication, authorization, auditing. The encryption provides the confidentiality and data integrity. Encryption transfer data into something an attacker cannot understand.

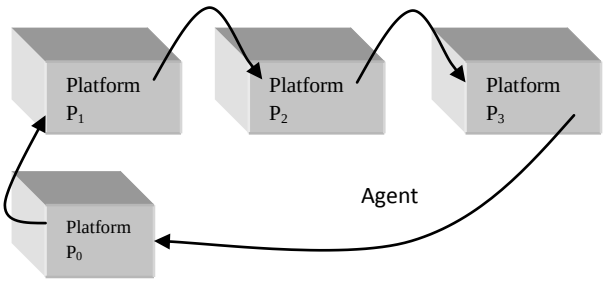

Figure 2 : Migration of Mobile Agent

The authentication is used to verify the claimed identity of a user, client, server, and so on. In the case of clients, the basic premise is that before a service will do work for a client, the service must learn the client's identity. Typically the users are authenticated by means of password that is password-based authentication, but there are many other ways to authenticate clients. After a client has been authenticated, it is necessary to check whether that client is authorized to perform the action requested.

5.1 Agent Travel Methods

Agent is dispatched from its origin and received by the specified destination. It is performed by an agent itself, another agent in the same place, or an agent or non-agent system outside of the place. Origin place contacts destination place and returns failure indication to agent if no contact. Destination must be known prior to departure; agent notifies the destination agent system it would like to depart. Destination agent system process request for departure Destination agent system suspends the agent, serialize the agent's data, determine the transport protocol for the agent, and transport the agent. Receiving engine determines to accept or reject and agent from the sending host and sender must authenticate itself to the receiving engine. Receiving engine receive the agent, decode its data information, deserialize the agent, sets new instantiation for the agent and resume agent execution.

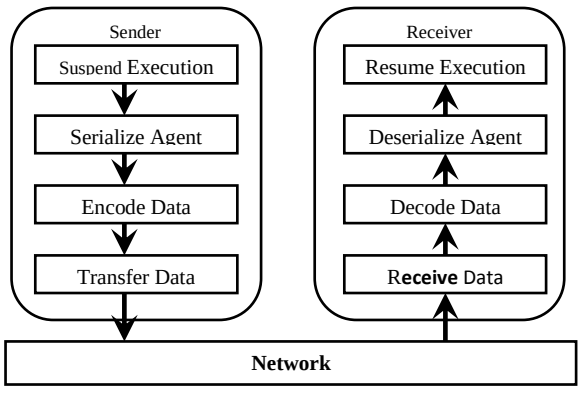

Figure 3: Travel of an Agent

5.2 Mobile Code Transfer Methods

The mobile code is transferred using one of the three methods class (or mobile code) destination, class at origin, code-on-demand. In the class at destination method, the engine's class is either in cache or the local files system and no need to transfer class, transferred agent must contain the full class name and discriminator and may have information that describes the location of the class definition. In the class at origin method the class is transported with the agent's state to the destination engine, state may transferred more than once. This technique can lead to increased network

traffic and wasted network bandwidth. In the code-on-demand method the class is made available by a server, destination engine receives class on a code-on-demand basis. In this method the destination address must perform an additional network connection.

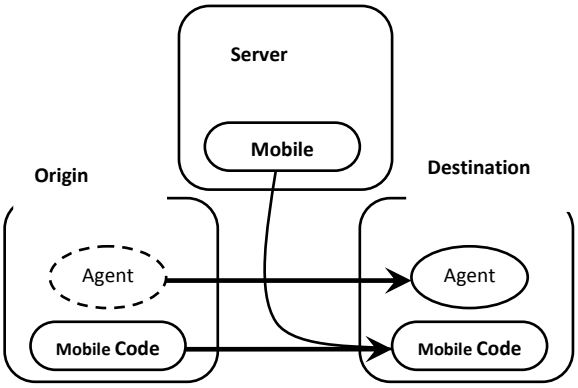

Figure 4: Mobile Code Transfer Options

5.3 Implementation Model of AMA

The authentication mechanism that uses in AMA has the some components. The protection starts by ensuring that handles the transfer of program to the client machine can be trusted. The AMA migrates from one machine to another machine of the agent-enabled network. When the authentication service is required in a distributed system on a particular machine, the system calls the proxy that enables the agent-enabled network.

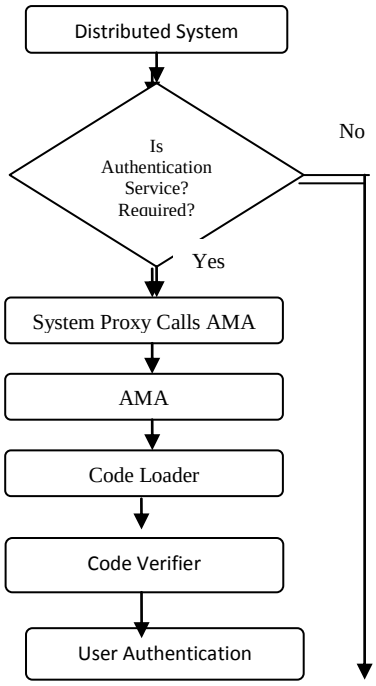

Figure 5: Flow of operations in AMA

The system proxy calls AMA. The code loader loads the mobile code which is used to verify the identity of the claimed entity on the destination machine. The mobile code is then verified by the code verifier in order to ensure the authentication of the mobile code itself. The code verifier uses DSA for generating the digital signature.

The AMA has a port number to listen the request from the machines. When the authentication service is required, the client process performs multicasts by sending UDP datagram with multicast address and the port number.

5.4 Authentication of the Mobile Code

The security issues of the mobile code are crucial. The component Code Verifier authenticates the mobile code. One of the methods of message authentication technique is digital signature. There are two approaches to create the digital signature, RSA approach and DSS approach. The standard is called DSS and algorithm DSA.

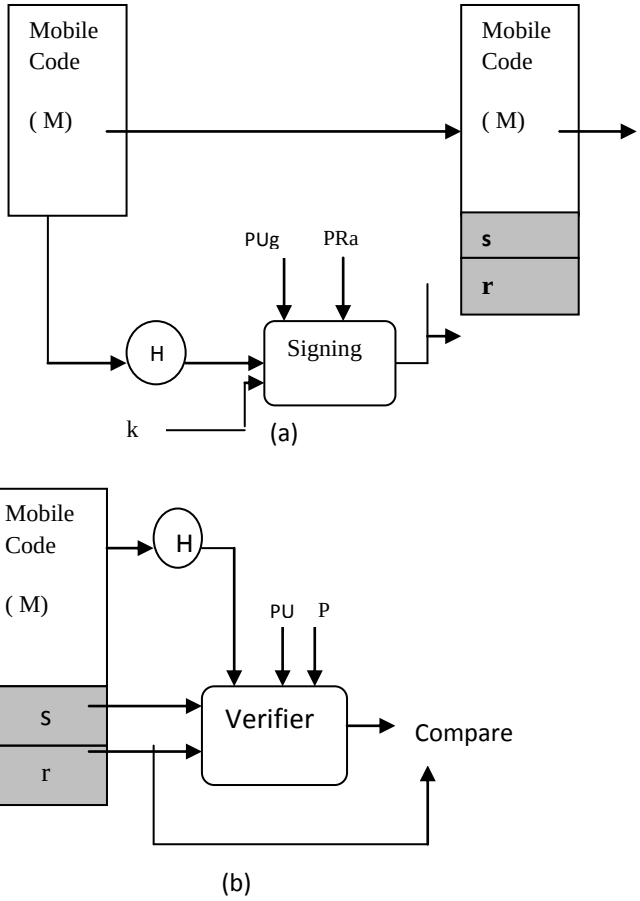

Figure 6: (a) Signing (b) Verifying of digital signature using DSA

The code verifier component of AMA implements the DSA for the authentication of the mobile code that is migrated in AMA. It is implemented using the standard java API. The DSA scheme takes four steps such as key generation, signing the message, computing the signature, and verifying the signature. These four steps using Java are described as follows.

5.4.1 Key generation

To make a new key pair, KeyPairGenerator object is used.

```
KeyPairGenerator kg = KeyPairGenerator.getInstance("DSA");
```

To generate keys, the key generation algorithm object is initialized with the key strength and a secure random number generator. The key strength is not the length of the generated keys but the size of one of the building blocks of the key. In the case of DSA, it is the number of bits in the modulus, one of the mathematical quantities that make up the public and

private keys. Suppose you want to generate a key with a modulus of 512 bits.

SecureRandom sr = new SecureRandom();

kg.initialize(512, sr);

Generating the key pairs:

KeyPair k1 = kg.generateKeyPair();

KeyPair k2 = kg.generateKeyPair();

Each key pair has a public and a private key.

PublicKey pubkey = k1.getPublic();

PrivateKey privkey = k2.getPrivate();

5.4.2 Signing the message

To sign a message, signature algorithm object of java Signature factory is used as given below.

Signature signalgorithm = Signature.getInstance("DSA");

The Signature algorithm objects can be used both to sign and to verify a message. Signature algorithm objects can be used both to sign and to verify a message. To prepare the object for message signing the initSign() method is used and pass the private key to the signature algorithm.

signalgorithm.initSign(privkey);

5.4.3 Computing the signature

The signature is computed using sign() method and returned it as an array of bytes.

byte[] signature = signalgorithm.sign();

The recipient of the message must obtain a DSA signature algorithm object and prepare it for signature verification by calling the initVerify() method with the public key as parameter.

Signature verifyalgorithm = Signature.getInstance("DSA");

verifyalgorithm.initVerify(pubkey);

5.4.4 Verifying the signature.

boolean v = verifyalgorithm.verify(signature);

If the verify method returns true, then the signature was a valid signature of the message that was signed with the matching private key. That is, both sender and the contents of the message have been authenticate

6. USER AUTHENTICATION

The DSA algorithm is used for generating the signature which is used to authenticate the mobile code. For user authentication for a particular service, AMA uses the password encrypted with the secret key which is attached in the mobile code that is downloaded by the calling process. The mobile code contains a database about the users with encrypted password. The AMA uses a password to authenticate ID of individual users. The ID determines whether the user is authorized to gain access to authentication service in a system, only those who already have an ID field on a system are allowed to request for the service. The key is used to encrypt password of each user. The DES is used to encrypt the password when loading it in a file. To verify the authenticate user for the service, system read password from the user and encrypt it using the secret key and compare it with the encrypted password stored in the file. If it matches, the user is authenticated for the requested services. Otherwise the user is not authenticated. Same process is performed in each machine. The Fig 7 illustrates how the password loads in a file and verify the password for user authentication.

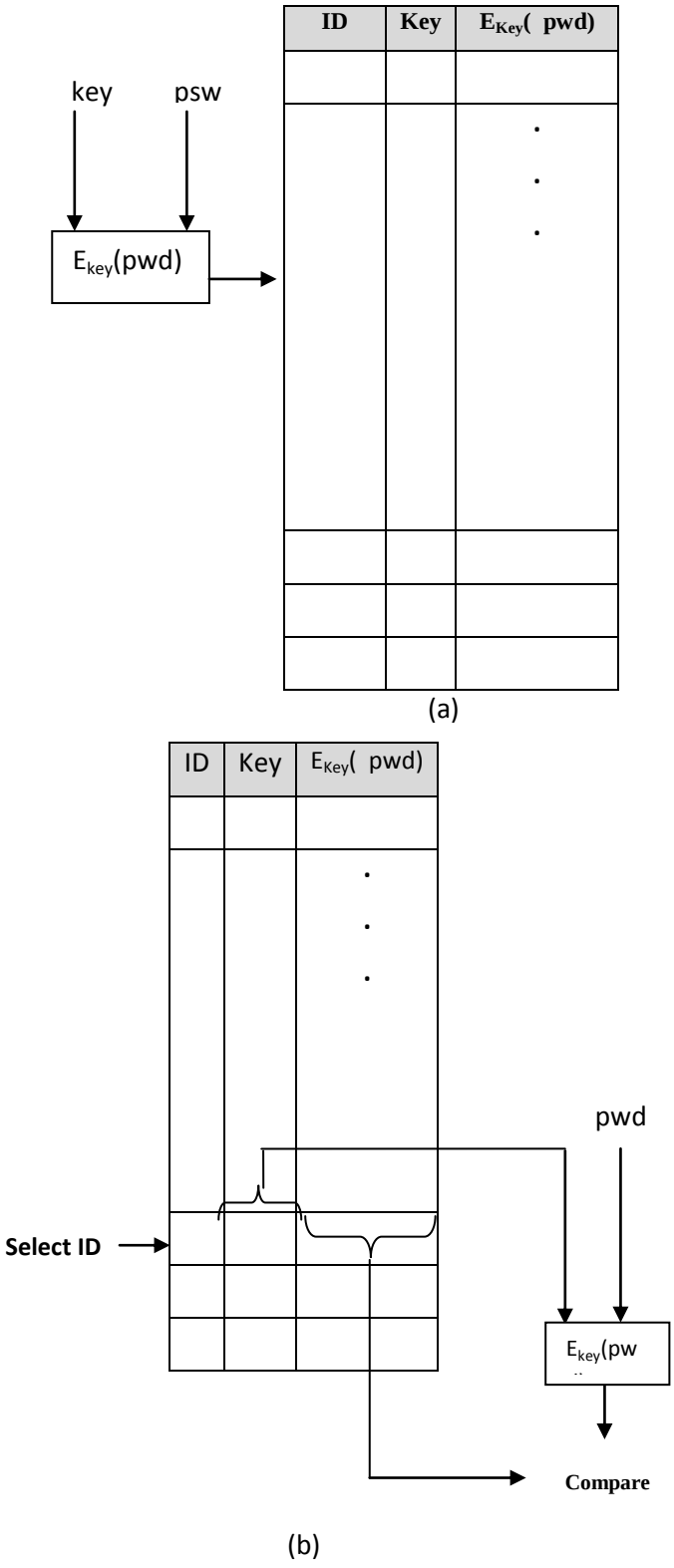

Fig 7: (a) Loading password (b) Verifying the password for user authentication.

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

In the mobile agent-based model, a number of agents are responsible for the authentication service. An agent is stored in each site and provides the authentication service by moving machine to machines. Such the agent is termed as an authentication mobile agent. Each AMA is controlled by an authentication server. The system maintains a table for each AMA. When the authentication server updates in the table, it informs to the AMA of particular site.

With a strong architectural foundation with standard security mechanisms, the service provided by the authentication mobile agent is efficient and effective for the distributed system. When the distributed system is scale up in number of sites the performance of the mobile agent based service is high as compare to the client server model service. Thus, it is concluded that the mobile agent-based services are effective and efficient rather than centralized controlled services for supporting the very high scalability and performance of the distributed systems. At each machine of the distributed system a user is authenticated by comparing the stored encrypted password with new password encrypted with a secret key.

7.2 Future Work

In this paper it is just describes an agent-based conceptual architectural model for providing user authentication in distributed systems. The mobile agent paradigm is the emerging field in the research. The future researches such as load balancing in parallel processing, distributed intrusion detection, monitoring the passive attacks, Monitoring the network traffic and congestion control, distributed deadlock detection using the mobile agent are possible.

8. REFERENCES

[1] Andrew S. Tanenbaum, Maarten van Steen “Distributed Systems: Principles and Paradigms”,PHI India 2002, ISBN 81-203-2215-0

[2] Wayne Jansen, Tom Karygiannis” Mobile Agent Security” NIST Special Publication

[3] David Chess, Colin Harrison, and Aaron Kershenbaum, “Mobile Agents: Are They are Good Idea?” IBM Research Report

[4] Syed Adnan, John Datuin and Pavana Yalamanchili “A Survey of Mobile Agent Systems”

[5] Jr. and Peter Gerken and Martin Hofmann and Daria Chacón and Greg Hill and Niranjan Suri. “Mobile-Agent versus Client/Server Performance: Scalability in an Information-Retrieval Task, Dept. of Computer Science, Dartmouth College, January, 2001

[6] David Kotz and Robert S. Gray “Mobile Agents and the Future of the Internet “,Department of Computer Science / Thayer School of Engineering Dartmouth College , 1999

[7] William Stallings “Network Security Essential: Applications and Standards”, Pearson Education, ISBN 81-7808-707-8

[8] Fuggetta , A, Ficco , G.P and Vigna G. “Understanding Code Mobility” IEEE Trans. Softw. Eng, Vol. 24, no. 5, pp 342-361, May 1998

[9] Bogdan C. Popescu, Bruno Crispo, Andrew S. Tanenbaum,Maas Zeeman “Expressing security policies for distributed objects applications”, Vrije Universiteit, Amsterdam, February 5, 2004.

[10] M. Van steen, P. Humburg, and A. Tanenbaum. Globe: A Wide-Area Distributed System. IEEE Concurrency, Pages 70-78, January-March 1999.

[11] Danny B. Lang and Mitsuru Oshima. Seven Good Reasons for Mobile Agents. Communication of ACM, March 1999