SECURING MULTI-AGENT BASED NETWORK MONITORING PLATFORM AGAINST MALICIOUS AGENT ATTACK

John-Otumu A. M[1*], Aliga P. A[2], Ojieabu C. E[3] and Akpe A. C. E[1]

[1]Directorate of Information & Comm. Technology, Ambrose Alli University, Ekpoma, Nigeria
[2]Department of Computer Science, Ambrose Alli University, Ekpoma, Nigeria
[3]Department of Electrical/Electronics, Ambrose Alli University, Ekpoma, Nigeria

ABSTRACT*:* *This research paper examined the security threat issues against agents/multi-agent based system platform by malicious agent attackers in a network environment. Several techniques like fault isolation or sandboxing, access control to host resources, digital signatures, strong authentication, proof carrying code and message encryption were suggested by different research scholars as a means of mitigating the menace but however, no strong evidence on their application / implementation were mentioned. This research work used a 2 Factor or Double Data Encryption Standard (DES) approach to encrypt / decrypt messages between agents in our proposed network monitoring platform to prevent malicious agent from hijacking the exact network data content during communication. Java programming language was used to implement the 2DES algorithm in the security mechanism proposed. The system was tested for its effectiveness and efficiency in both non-production and production network environment. Results revealed detection rate to be high and with a very minimal false alarm rate. The proposed system is highly recommended for usage in any local area network environment.*

KEYWORDS: 2DES Algorithm, Java, Agents, Security, LAN

INTRODUCTION

Network management software solutions should have mechanisms to monitor, control and coordinate the Open System Interconnection (OSI) environment's resources for information exchange between resources in the areas of Fault, Configuration, Accounting, Performance and Security (FCAPS). According to (Islam and Taj-Eddin, 2012) a network monitoring system is seen as a gatekeeper for checking all incoming and outgoing connections or traffic, security metrics, link status, and latency in the network environment.

It is noted from literature reviewed that the initial network management system were faced with some of the following problems: static, centralized and polling-based system that involves high capacity computing resources at centralized platform in which the network traffic might oppress the network bandwidth (Abar *et al*, 2008; Carvalho and D'mello, 2013), overwhelming volume and complexity of the information involved in network management that may impart a terrible load on the centralized system (Consens and Hasan, 1993; Ismail *et al*, 2000), high network latency and lack of fault tolerance (Carvalho and D'mello, 2013), huge bulk of traffic that remains concentrated around the centralized station (Stephen *et al*, 2003; Carvalho and D'mello, 2013).

The efficient management of a computer network is now a critical issue in today's rapid changing network environment. Though several research scholars in the field of agent technology had used the strength of agent and multi-agent based systems to develop network monitoring applications with the bid of finding solutions to the initial challenges faced as stated above; but they have also been bedeviled by control, coordination and interoperability issues thereby making the system incapable of executing all the network monitoring services (Fault, Configuration, Accounting, Performance and Security) concurrently in a single automated platform (John-Otumu *et al*., 2017) as recommended by (Akinyokun *et al*., 2014) in order to make the network administrator more efficient and proactive.

Software agent is defined as a computer system situated in an environment with the ability of taking autonomous actions in order to meet its desired goals (Akinyokun *et al*., 2014). Software agents can also be seen as secret software detectives that provide a platform for computing and execution of various tasks like information gathering, information filtering, searching, online shopping, and personal assistant (Chang *et al*., 2011).

Software agents can exist as a single entity or in collaboration with other agents in the same environment to carry out some specific task which can be referred to as multi-agent system or as a mobile agent that has extra mobility function to move from one node in a computer network to another; while multi-agent system is concerned with a system or collective behavior of existing autonomous agents in some kind of environment aiming at proffering solutions to a given problem.

Jansen (2002) is also of the opinion that a multi-agent based system can be seen as a system that is a loosely coupled network troubleshooter or solution provider that works together to solve issues that is beyond the capability of an individual agent.

Now, the ongoing explosion in the usage of multi-agent based applications for the monitoring or management of network resources with the bid of solving issues has completely revolutionized the computer network environment.

However, this explosion is introducing a huge problem of security threat for both the networks and the agent services. The main objective of this research paper is to provide a solution for protecting multi-agent system communicating within a platform from malicious agent trying to intercept and destroy sensitive network data (Karnik and Tripathi, 2000) using encryption standards (Zhou *et al*., 2014).

This research paper intends to fill the gap created by using a 2 factor DES. Data Encryption Standard (DES) is a standard for encrypting data using a symmetric algorithm like Data Encryption Algorithm (DEA). It is a block cipher developed by the IBM cryptography researcher Horst Feistel in the early 1970's. It consists of a number of rounds where each round contains bit-shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. DES expects two inputs - the plaintext to be encrypted and the secret key. The manner in which the plaintext is accepted and the key arrangement used for encrypting and decrypting data is both determined by the type of cipher used. The DES is therefore a symmetric, 64 bit block cipher as it uses the same key for both encryption and decryption and only operates on 64 bit blocks of data at a time.

LITERATURE REVIEW

There is a general security issues with agents since they tend to operate at the background and autonomously carry out their task. They are sometimes seen as threats to other software and also sometimes seen as threat amongst agents thereby competing for similar resources within the computer system / network or killing themselves. According to (Chess, 1998) mobile code system security is a hard task to achieve because it negates the basic facts that exist in most computer security measures.

Key security properties of agents

According to (Jung, 2012) agent based systems needs the following security properties:

(a) Confidentiality

Confidentiality is the act of ensuring that the agent is able to keep and make private data personal to itself or the agent platform. In other words, no unauthorized persons or programs should be allowed to read or alter the data without due permission. Different security models have been proposed for the purpose of confidentiality protection of agents; Code obfuscation (Armoogum and Caully, 2010), environmental key generation, access control to data and programs, encryption (Ssekibuule, 2010).

(b) Integrity

Agent complies with security requirement which ensures that the agent's code, state and data must not be changed or updated by an unauthorized agent or platform; in other words, only an authorized entity should have the right and privilege to modify the agent contents. Software agents can only detect the presence of data alteration, but can hardly stop other malicious agents or platform from changing its state, code or data.

Several researchers have proposed different schemes for the protection of agent components. Farmer *et al* (1996) proposed an architectural model for trust relationship between principal agent systems. The designed architectural model was able to shield users and host from attacks through agent modification. Cao and Lu (2005) proposed an access control architecture which captures the path history of the agent. Mitrovic and Arribalzaga (2002) also proposed an architectural model for securing agent based systems using trusted domain and proxy agents. The model ensured transparent and secured services to both security ware agents and legacy agents.

(c) Accountability

Accountability in agent-based systems is about ensuring that the entities of agent comprising user, process host and the agent itself is accountable for their activity and they must be uniquely identified, authenticated and audited.

(d) Availability

Availability is a vital security requirement of agent based systems. The agent host ensures the availability of data and services for the remote and local agents. Agent platforms possess the following characteristics to be able to communicate.

i. Deadlock management

ii. Concurrent access

iii. Concurrency control

iv. Restricted access

v. Capacity for agent dispatching

(e) Non-repudiation

Repudiation will occur in agent-based systems when an agent that participated in a given transaction later claimed that the transaction never occurred. Agent-based systems must try to ensure that proper measures are put in place to prevent non-repudiation among communicating entities.

ii. Agent-based security threats and protection strategies

Software agents (static or mobile) can be of security threats to themselves or the hosts / platforms in which they operate on. Agent-based security threats can be broadly classified as follows:

(a) Agent-to-Agent threats

(b) Agent-to-Platform threats

(c) Platform-to-Agent threats

Though mobile agent functionality has many advantages; its ability to automatically execute arbitrary codes from one host to another on the network can create a lot of security threats for both the mobile agent and the host. According to (Reiser and Vogt, 2000) mobile agents must be able to access and configure security-sensitive resources, hence, it is possible for the mobile agent to leak or destroy sensitive data and cause the host to function abnormally.

Conversely, mobile agents also need to protect them against hostile or malicious host, as they may sometimes be carrying sensitive data with them which the host might compromise or destroy (Karnik and Tripathi, 2000).

(a) Agent-to-Agent threats

These are threats that enable agent-based systems to exploit security vulnerabilities of other agents in order to lunch attack against the agents on their own platform. This type of threats includes Denial of Service (DoS) attack, repudiation, unauthorized access and masquerading. According to Mishra and Choudhary (2012) protection against this type of attack is done by the implementation of separate agents on agent host.

(b) Agent-to-Platform threats

This form of threat occurs when a malicious agent attacks an agent platform. This type of attack can take the form of denial of service (DoS), masquerading, and unauthorized access to the host resources. However, from literatures reviewed, agent’s platforms could be protected against malicious agent attack in the following ways:

(i) Fault isolation or Sandboxing (Mao *et al*., 2011)

(ii) Access control to host resources (Reeja, 2012; Zhou *et al*., 2014)

(iii) Digital signatures (Madhulika and Rao, 2014)

(iv) Strong authentication (Shin *et al*., 2014)

(v) Proof carrying code (Pirzadeh *et al*., 2010)

(vi) Message encryption (Zhou *et al*., 2014)

(c)Platform-to-Agent threats

This is an attack from a platform against an agent. The set of threats under this type of attack includes the following; repudiation, denial of service (DoS), alteration, eavesdropping, masquerading, migration of agent, and killing of agent as proposed by some researchers like (Chang *et al*., 2011; Farmer *et al*., 1996; Mishra and Choudhary 2012; Madhulika and Rao 2014).

METHODOLOGY

Here, we used a 2 factor Data Encryption Standard (DES) algorithm to design the security agent mechanism that will be able to create a secure end-to-end communication process between the client agent and the server agent controlling the major activities on the platform in order to protect the data carried by the mobile agent from one host to another or being transmitted between the server agent and node or client agent vice versa in the network environment so that information being sent or received by the agents can be encrypted or decrypted because it is much faster computationally as compared to other asymmetric encryption. The blueprint was implemented using Java programming language because of its abstraction features. The interactivity of the system is also modeled used a sequenced diagram.

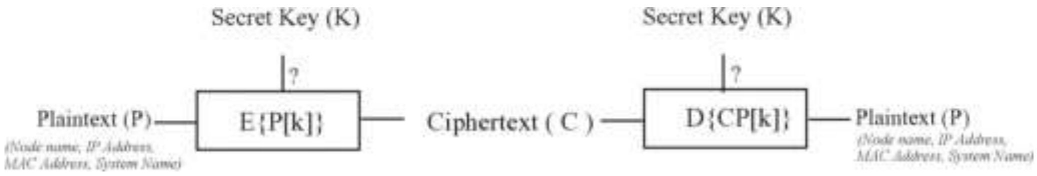

Secret Key (K)
?
Plaintext (P)
(Node name, IP Address, MAC Address, System Name)
E{P[k]}
Ciphertext (C)
Secret Key (K)
?
D{CP[k]}
Plaintext (P)
(Node name, IP Address, MAC Address, System Name)

Figure 1: Encryption/decryption process Adapted from (Babar and Bhope, 2016)

Figure 1 shows the graphical form of the encryption / decryption process. Our security agent was designed to use the 2DES algorithm to mathematically transform data from plaintext to cipher text and from cipher text to another cipher text. We used four basic input parameters in profiling each node on the network (Node name, IP Address, MAC Address, and System name). This secured communication process is necessary in order to avoid theft or eavesdrop of data by any malicious agents or attacker. Each node's data profiled is always being synchronized between the server and client agents for update information from both ends since a network environment is dynamic. Malicious agents can position themselves between the server and client agent to steal data during this update process for their personal benefits. When this occurs all they can see will be scrambled data that might be useless to them at that particular point in time.

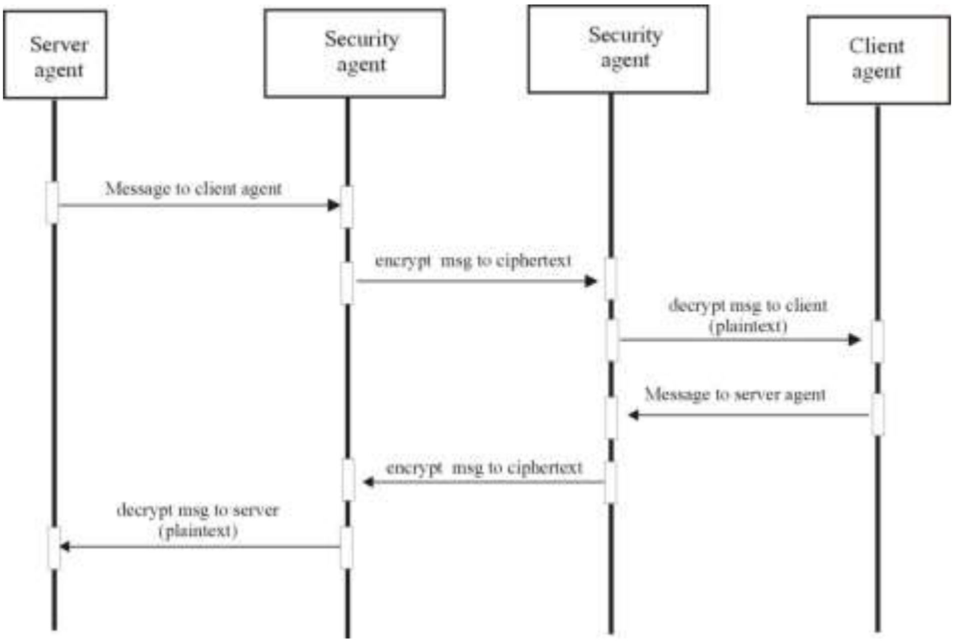

Server agent
Security agent
Security agent
Client agent
Message to client agent
encrypt msg to ciphertext
decrypt msg to client (plaintext)
Message to server agent
encrypt msg to ciphertext
decrypt msg to server (plaintext)

Figure 2: Modeling Secure End-to-End Encryption/Decryption between Server and Client Agent

Figure 2 shows the interaction between the server, security and client agents to establish a secure end-to-end communication process using Data Encryption Standard (DES) algorithm.

RESULTS AND DISCUSSION

The analysis and design carried out on securing a multi-agent platform against malicious agent theft or total destruction of network data using a 2 factor Data Encryption Standard (DES) as agent to agent authentication means is further developed into a software prototype.

This section provides a documented basis for ensuring that the proposed system will perform according to its features and functions as required. It ensures that the system output is in line with the main objective of the research work; and this plays an important role of verification and validation in software development.

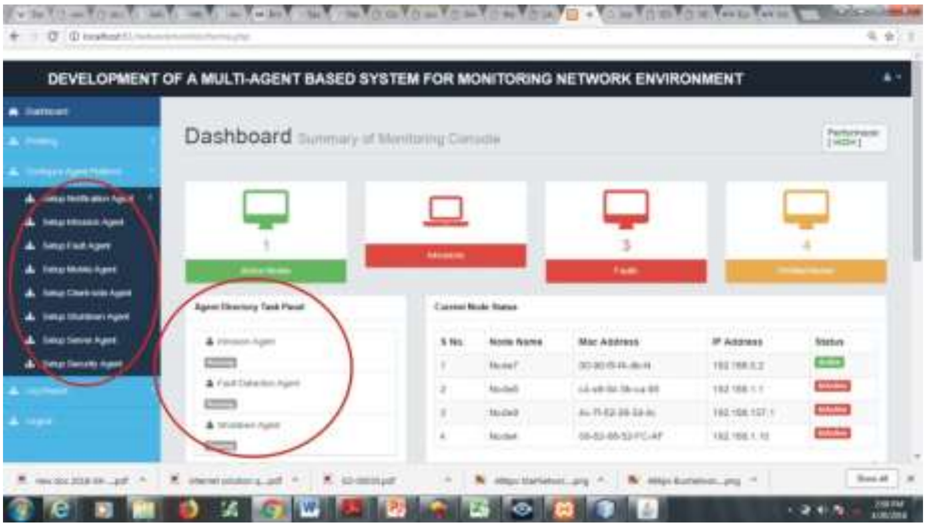


Figure 3: Developed Multi-agent network monitor dashboard

Figure 3 shows the developed multi-agent based network monitoring dashboard. The two red circled areas of the dashboard further revealed all the agents currently running in the agent's directory task panel, and all the agents that can be setup (activated/deactivated) on the configure agent platform.

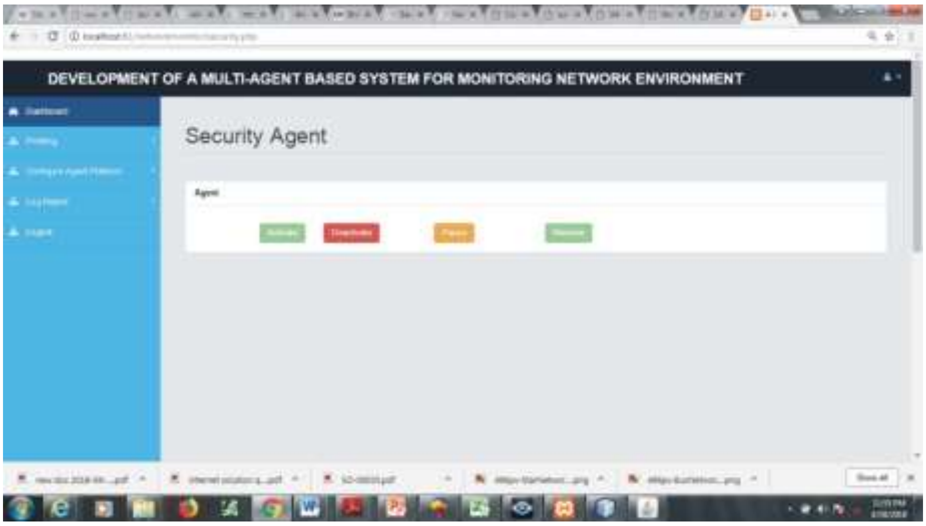
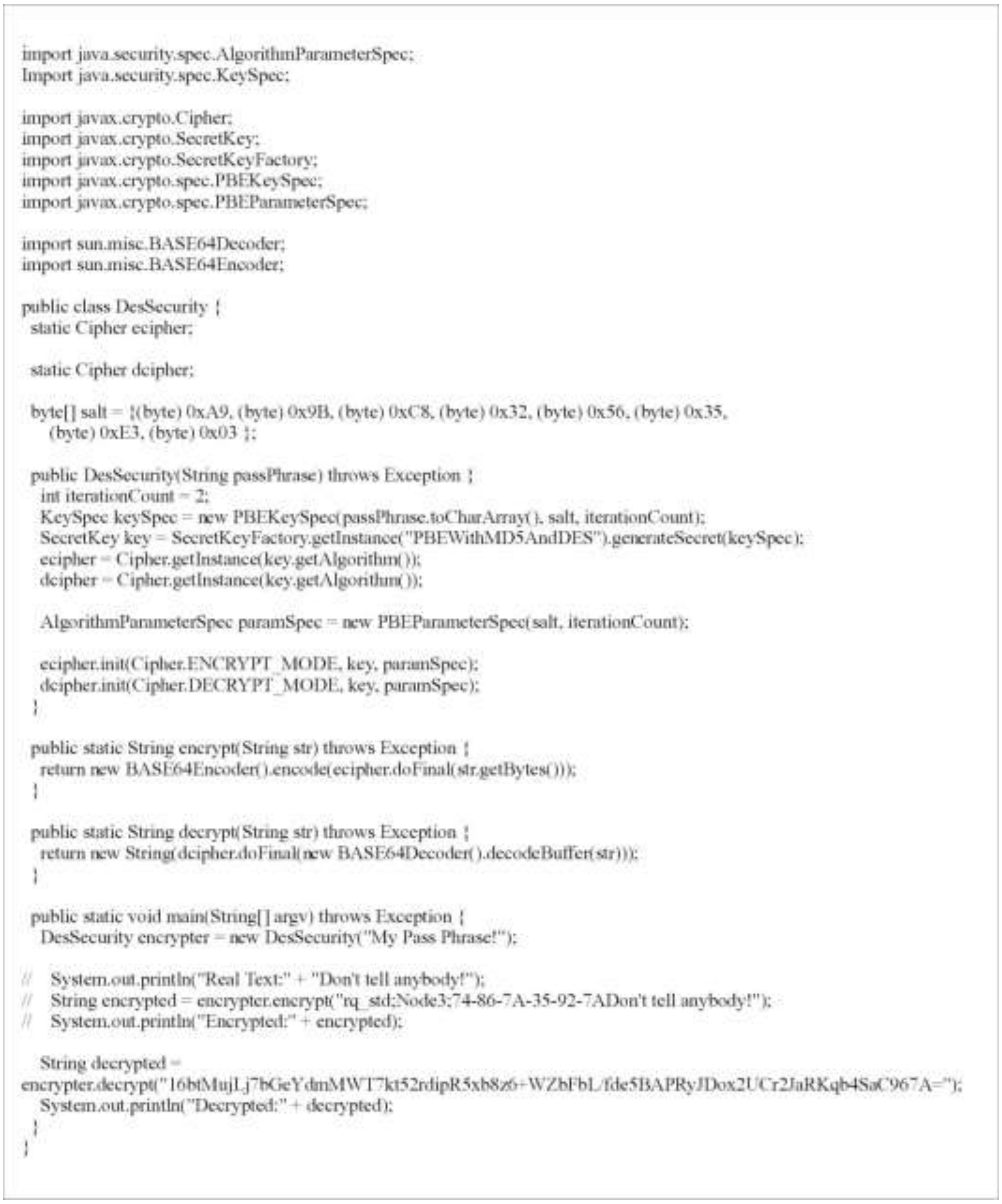

DEVELOPMENT OF A MULTI-AGENT BASED SYSTEM FOR MONITORING NETWORK ENVIRONMENT
Security Agent
Agent

Figure 4: Interface for activating/deactivating security agent

Figure 4 shows the interface for activating / deactivating the security agent of the developed multi-agent based network monitoring system. Once activated the agent starts running i.e. encrypting / decrypting data between the node/client agents and the server agent through the designed authentication scheme (2DES) as supported by (Zhou *et al.*, 2014; Reeja, 2012; Zhou *et al.*, 2014; Madhulika and Rao, 2014; Shin *et al.*, 2014; Pirzadeh *et al.*, 2010), until whenever the network administrator deactivates it to normal state.



```
import java.security.spec.AlgorithmParameterSpec;
Import java.security.spec.KeySpec;

import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.PBEParameterSpec;

import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;

public class DesSecurity {
  static Cipher ecipher;

  static Cipher dcipher;

  byte[] salt = {(byte) 0xA9, (byte) 0x9B, (byte) 0xC8, (byte) 0x32, (byte) 0x56, (byte) 0x35,
     (byte) 0xE3, (byte) 0x03 };

  public DesSecurity(String passPhrase) throws Exception {
    int iterationCount = 2;
    KeySpec keySpec = new PBEKeySpec(passPhrase.toCharArray(), salt, iterationCount);
    SecretKey key = SecretKeyFactory.getInstance("PBEWithMD5AndDES").generateSecret(keySpec);
    ecipher = Cipher.getInstance(key.getAlgorithm());
    dcipher = Cipher.getInstance(key.getAlgorithm());

    AlgorithmParameterSpec paramSpec = new PBEParameterSpec(salt, iterationCount);

    ecipher.init(Cipher.ENCRYPT_MODE, key, paramSpec);
    dcipher.init(Cipher.DECRYPT_MODE, key, paramSpec);
  }

  public static String encrypt(String str) throws Exception {
    return new BASE64Encoder().encode(ecipher.doFinal(str.getBytes()));
  }

  public static String decrypt(String str) throws Exception {
    return new String(dcipher.doFinal(new BASE64Decoder().decodeBuffer(str)));
  }

  public static void main(String[] argv) throws Exception {
    DesSecurity encrypter = new DesSecurity("My Pass Phrase!");

//    System.out.println("Real Text:" + "Don't tell anybody!");
//    String encrypted = encrypter.encrypt("rq_std;Node3;74-86-7A-35-92-7ADon't tell anybody!");
//    System.out.println("Encrypted:" + encrypted);

    String decrypted =
encrypter.decrypt("16btMujLj7bGeYdmMWT7kt52rdipR5xb8z6+WZbFbL/fde5BAPRyJDox2UCr2JaRKqb4SaC967A=");
    System.out.println("Decrypted:" + decrypted);
  }
}
```

Figure 5: Java code segment for implementing the 2DES Algorithm

Figure 5 shows the Java programming language code segment used to implement the 2DES algorithm in the security agent mechanism.

CONCLUSION

The problem of effective multi-agent based network management software is an interesting and challenging one. The security threats amongst agents in a multi-agent based platform and that of a malicious agent attacking other agents or agent's platform is another big challenge. This research paper addressed the issue of a malicious agent attack on a multi-agent based platform using a 2 Factor or Double DES algorithm.

However, a multi-agent based network monitoring software prototype designed and developed was able to take care of the initial security gap in knowledge created.

Future Research Work

We recommend that future research work should consider the implementation of multiple certificates authentication scheme for communicating agents within a platform in order to prevent theft or destruction of valuable network data by malicious agents.

REFERENCES

Abar, S., Konno, S., and Kinoshita, T. (2008). Autonomous Network Monitoring System based on Agent-mediated Network Information, *International Journal of Computer Science and Network Security*. 8(2), 326 – 333.

Akinyokun, O. C., Ekuewa, J. B., and Arekete, S. A. (2014). Development of Agent-Based System for Monitoring Software Resources in a Network Environment, *Artificial Intelligence Research*, 3(3). 62 – 74.

Armoogum, S. and Caully, A. (2010). Obfuscation Techniques for Mobile Agent Code Confidentiality, *Journal of E-Technology*, 1(2), 83-94.

Babar, P. K., and Bhope, V. P. (2016). Design and Implement Dynamic Key Generation to Enhance DES Algorithm. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 4(7), 465 – 472.

Cao, C. and Lu, J. (2005). A Path-History-Sensitive Access Control Model for Mobile Agent Environment. *In Distributed Computing Systems Workshops, 2005. 25th IEEE International Conference*.

Carvalho, L., and D'mello, N. (2013). Secure Network Monitoring System Using Mobile Agents, *International Journal of Modern Engineering Research (IJMER)*. 3(3), 1850 – 1853.

Chang, Y. S., Yang, C. T., and Luo, Y. C. (2011) An Ontology based Agent Generation for Information Retrieval on Cloud Environment. J. UCS, 17(8), 1135-1160.

Chess, D. M. (1998). Security Issues in Mobile Code Systems, *Mobile Agents and Security*, 1419, 1 – 14.

Consens, M., and Hasan, M. (1993). Supporting Network Management through Declaratively Specified Data Visualizations. *In IEEE / IFIP 3rd International Symposium on Integrated Network Management*, 725 – 738.

Farmer, W. M., Guttman, J. D., and Swarup, V. (1996) *Security for mobile agents: Authentication and State Appraisal*. *Computer Security—ESORICS 96*. Springer.

Islam, A. T. F., and Taj-Eddin (2012). A .Net Framework Approach for a Network Monitoring Tool, *International Journal of Computer applications*,55(10), 1 – 14.

Ismail, L., Hagimont, D. and Mossiere, J. (2000). Evaluation of the Mobile Agents Technology: Comparison with the Client/Server Paradigm, *Information Science and Technology (IST)*, 19(1).

Jansen, W. (2002). Intrusion Detection with Mobile Agents, *Computer Communications*, 25(15),1392 – 1401.

John-Otumu, A. M., Ojieabu, C. E., and Oshoiribhor, E. O. (2017). An Enhanced Network Monitoring System using Multi-Agent based Technology, *International Journal of Advanced Research in Computer and Communication Engineering*, 6(5), 1 – 8.

Jung, Y. (2012). A Survey of Security Issue in Multi-Agent Systems. *Artificial Intelligence Review*, 37(3), 239 - 260.

Karnik, N. M., and Tripathi, A. R. (2000). A Security Architecture for Mobile Agents in Ajanta, *Proceedings of 20th International Conference on Distributed Computing Systems*, Taipei, Taiwan, 402 – 409.

Madhulika, G. and Rao, C. S. (2014). Generating Digital Signature using DNA Coding. *In Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA)*, Springer.

Mao, Y., Chen, K. T., and Srinivasan, D. (2011). Software Fault Isolation with API Integrity and Multi-Principal Modules, *In Proceedings of the 23rd ACM Symposium on Operating Systems Principles*.

Mishra, A. and Choudhary, A. (2012). Mobile Agent: Security Issues and Solution, *International Journal of Computer Technology and Electronics Engineering (IJCTEE)*, 2(3), 54 - 75.

Mitrovic, N. and Arribalzaga, U. A. (2002). Mobile Agent Security Using Proxy-Agents and Trusted Domains. *In 2nd International Workshop of Security of Multiagent Systems (SEMAS'02) at 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 02)*, DFKI Research Report.

Pirzadeh, H., Dubé, D., and Hamou-Lhadj, A. (2010). An extended Proof-Carrying Code Framework for Security Enforcement, *In Transactions on computational science*, 11, 249 - 269.

Reeja, S. (2012). Role Based Access Control Mechanism In Cloud Computing Using Co–Operative Secondary Authorization Recycling Method, *International Journal of Emerging Technology and Advanced Engineering*, 2(10), 240 – 250.

Reiser, H., and Vogt, G. (2000). Threat Analysis and Security Architecture of Mobile Agent Based Management Systems, *Network Operations and Management Symposium (NOMS)*, Honolulu, HI. 979 - 980.

Shin, E. A., Beygelzimer, A., Grabarnik, G., and Hernandez, K. (2014). An Effective Authentication Mechanism for Ubiquitous Collaboration in Heterogeneous Computing Environment. *Peer-to-Peer Networking and Applications*, 7(4), 612 - 619.

Ssekibuule, R. (2010). Mobile Agent Security against Malicious Platforms, *Cybernetics and Systems: An International Journal*, 41(7), 522 - 534.

Stephen, R., Ray, P., and Paramesh, N. (2003) Network Management Platform Based On Mobile Agent. *International Journal of Network Management*, 14, 59 -73

Zhou, L., Varadharajan, V., and Hitchens, M. (2014). Cryptographic Role-Based Access Control for Secure Cloud Data Storage Systems, *In Security, Privacy and Trust in Cloud Systems*. Springer, 313 - 344.