

## **ABSTRACT**

In view of the security uncertainty on the Internet, network intrusion detection technology can discover the intrusion behavior of the network and provide security protection for the security of the network. The network intrusion detection system automatically matches the IP of the machine. After the matching is successful, the display interface is divided into three modules: one-click scanning, whitelisting, and recording. Realized functions: It can monitor network= scans based on whitelists, and add and delete whitelists; unauthorized access can record and alarm, allowing users to choose whether to permit unauthorized access; can record relevant information of scanning, Unauthorized access is to distinguish between tags.

## **List of Abbreviations**

IDS - Intrusion detection System

IPS - Intrusion prevention system

IMS - Intrusion Management System

CNN - Convolutional Neural Networks

RNN - Recurrent Neural Networks

SRS - Software Requirement Specification

APPN -Advanced Peer-to-Peer Networking

## TABLE OF CONTENTS

CONTENT	PAGE NUMBER
1. Abstract and Abbreviations	7-8
2. Introduction	9
3. Literature Survey	10-13
4. System Analysis	14-15
5. System Requirement Analysis	16-24
6. Design	25-30
7. Implementation	31-35
8. Testing	36-40
9. Result and Analysis	41
10. Conclusion and Future Use	42
11. References	43
12. Annexure	44-52
13. Screenshots	53-77

# **CHAPTER 1**

## **INTRODUCTION**

At present, the new trend of intrusion detection technology (IDS) development is mainly in two directions, one is the tendency to build an intrusion prevention system (IPS). IPS is implemented by adding active response function in IDS, and accessing the network in series (IDS is connected to the network in parallel). Once an attack behavior is found, it responds immediately and actively disconnects the attacker. IPS not only has intrusion detection capabilities, but also has security protection features, which tend to build an Intrusion Management System (IMS). IMS is another direction for the development of IDS. The goal of IMS is to integrate multiple functions such as intrusion detection, vulnerability analysis, and intrusion prevention into one platform for management. IMS technology is a management process. When no attack occurs, IMS mainly considers the vulnerability information in the network, evaluates and judges the possible attacks and threats to be faced: in the event of an attack or an impending attack, not only the intrusion is detected. Active response and defense against intrusion behaviors. After being attacked, we must further analyze the intrusion behavior and use correlation analysis to determine the next possible attack.

The development trend of future intrusion detection systems is intelligent, management and security. It can be seen from the test results at home and abroad that the monitoring technology of the current network intrusion monitoring system is still not smart enough. First, it should be that application layer protocol support is not enough. Unable to process other applications. Second, the advancement of active defense technology. Even if there is an intrusion detection system called abnormal mode judgment, the impact on the unknown intrusion method is small, and the actual application effect of the new attack mode is not obvious. Third, combine network security technologies with network security and e-commerce technologies to provide complete network security.

## CHAPTER 2

### LITERATURE SURVEY

This chapter gives the overview of literature survey. This chapter represents some of the relevant work done by the researchers.

Many existing techniques have been studied by the researchers on network intrusion detection is done, few of them are discussed below.

**Topic:** Design and Implementation of Network Intrusion Detection System

**Author:** Z. Mu, H. Liu and C. Liu,

**Publication:** 2020 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Vientiane, Laos, 2020, pp. 494-497, doi: 10.1109/ICITBS49701.2020.00107.

**Technology Used:**

In view of the security uncertainty in the Internet, network intrusion detection technology can discover the intrusion behavior of the network and provide security protection for the security of the network. The network intrusion detection system automatically matches the IP of the machine. After the matching is successful, the display interface is divided into three modules: one-click scanning, whitelisting, and recording. Realized functions: It can monitor network scans based on whitelists, and add and delete whitelists; unauthorized access can record and alarm, allowing users to choose whether to permit unauthorized access; can record relevant information of scanning, Unauthorized access is to distinguish between tags.

**Topic:** Intrusion Detection in secure network for Cybersecurity systems using Machine Learning and Data Mining

**Author:** H. Azwar, M. Murtaz, M. Siddique and S. Rehman

**Publication:** 2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS), Bangkok, Thailand, 2018, pp. 1-9, doi: 10.1109/ICETAS.2018.8629197.

**Technology Used:**

The emergent rate of security dangers in the network commands extremely consistent well-being solution. Researchers usually worked on several way out to detect invasions. The security phases of intrusion detection using machine learning approach have been deliberated in our paper. In the meantime, Intrusion Detection (IDSs) played a crucial part in the outline and evolvement of stout linkage frame that basically secure system by distinguishing and intercepting multiplicity of assaults. A lot of techniques have been established that are built on machine learning approaches. Though, they are not exact effective in detecting all kinds of infringements. In our paper, a comprehensive analysis of several machine learning techniques has been supported for discovering the basis of glitches related with various machine learning techniques in perceiving invasive activities. Limitations accompanying with each of them are also discoursed. Several data mining tools for machine learning have also been contained within the paper. Consistent standard datasets happens serious to asses and estimate enactment of a detection structure. There subset many datasets, for example, DARPA98, KDD99, ISC2012, and ADFA13 etc. are used to estimate the performance of intrusion detection tactics but we have used the latest one in our research i.e. CICIDS2017 provided much better accuracy. Within this paper we commenced a broad assessment of the current datasets by means of our own projected standards, and put forward an estimation outline for IDS datasets. We upkeep this privilege by ascertaining challenges specific to network intrusion detection, and offer a set of guiding principle destined to build up future research on anomaly detection.

**Topic:** Using Deep Learning Techniques for Network Intrusion Detection

**Author:** S. Al-Emadi, A. Al-Mohannadi and F. Al-Senaïd

**Publication:** 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), Doha, Qatar, 2020, pp. 171-176, doi: 10.1109/ICIoT48696.2020.9089524.

**Technology Used:**

In recent years, there has been a significant increase in network intrusion attacks which raises a great concern from the privacy and security aspects. Due to the advancement of the technology, cyber-security attacks are becoming very complex such that the current detection systems are not sufficient enough to address this issue. Therefore, an implementation of an intelligent and effective network intrusion detection system would be crucial to solve this problem. In this paper, we use deep learning techniques, namely, Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) to design an intelligent detection system which is able to detect different network intrusions. Additionally, we evaluate the performance of the proposed solution using different evaluation matrices and we present a comparison between the results of our proposed solution to find the best model for the network intrusion detection system.

**Topic:** An Ensemble-based Network Intrusion Detection Scheme with Bayesian Deep Learning

**Author:** J. Zhang, F. Li and F. Ye

**Publication:** ICC 2020 - 2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 2020, pp. 1-6, doi: 10.1109/ICC40277.2020.9149402.

**Technology Used:**

Network intrusion detection is the fundamental of the Cybersecurity which plays an important role in preventing the systems away from malicious network traffic. Recent Artificial Intelligence (AI) based intrusion detection systems provide simple and accurate intrusion detection compared with the conventional intrusion detection schemes, however, the detection performance may not be reliable because the models in the AI algorithms must output a prediction result for each incoming instance even when the models are not confident. To tackle the issue, we propose to adopt Bayesian Deep Learning, specifically, Bayesian Convolutional Neural Network, to build intrusion detection models. Moreover, an ensemble-based detection scheme is further proposed to enhance the detection performance. Two open datasets (i.e., NSL-KDD and UNSW-NB15) are used to evaluate the proposed schemes. In comparison, Convolutional Neural Network and Support Vector Machine are implemented as baseline IDS (i.e., CNN-IDS and SVM-IDS). The evaluation results demonstrate that the proposed BCNN-IDS can significantly boost the detection accuracy and reduce the false alarm rate by adopting the proposed T-ensemble detection scheme.

**INFERENCE FROM STUDY**

From the above literature study, the techniques used are network intrusion methods, which requires large volume of data to be handled. Whereas the network intrusion avoidance is not studied with new techniques, which can be fulfilled by the proposed work.



## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **PROBLEM DEFINITION**

Consider Nodes A and B communicates via a wireless link. Within the communication range of both A and B there is an intruder node J. When A transmits a packet m to B, node J classifies m by receiving only the first few bytes of m. J then corrupts m beyond recovery by interfering with its reception at B. Addressed the problem of preventing the intrusion node from classifying m in real time, thus mitigating J's ability to perform selective attack.

#### **PROJECT OBJECTIVES**

The goal of a network intrusion detection system is to discover unauthorized access to a computer network by analysing traffic on the network for signs of malicious activity. The intrusion detection task is to build a predictive model capable of distinguishing between intrusions or attacks, and normal network connections.

#### **EXISTING SYSTEM:**

Intrusion detection systems are classified as network based, host based, or application based depending on their mode of deployment and data used for analysis. Additionally, intrusion detection systems can also be classified as signature based or anomaly based depending upon the attack detection method. The signature-based systems are trained by extracting specific Patterns (or signatures) from previously known attacks while the anomaly-based systems learn from the normal data collected when there is no anomalous activity.

#### **Disadvantages**

- ☐ Gives false alarms
- ☐ Fails to detects most of the attacks
- ☐ Slow in decision making, thus network traffic occurs

## **PROPOSED SYSTEM:**

The intrusion detection technique can be effective in improving the attack detection accuracy by reducing the number of false alarms, while the different approach can be implemented to improve the overall system efficiency. Hence, a natural choice is to integrate them to build a single system that is accurate in detecting attacks and efficient in operation.

Three schemes have been developed that prevent classification of transmitted packets in real time. These schemes rely on the joint consideration of cryptographic mechanisms with PHY-layer attributes.

### **Advantages**

- ☐ Gives very few false alarms
- ☐ Detects most of the attacks
- ☐ Copes with large amount of data
- ☐ Fast enough to make Real-time decisions
- ☐ System is accurate in detecting attacks

## **SYSTEM REQUIREMENT SPECIFICATIONS**

### **4.1. INTRODUCTION**

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.)

Under requirement specification, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

### **4.2. Requirement Analysis**

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium through which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

#### **4.2.1.1. Product Perspective**

The application is developed in such a way that any future enhancement can be easily implementable. The project is developed in such a way that it requires minimal maintenance. The software used are open source and easy to install. The application developed should be easy to install and use.

#### **4.2.1.2. Product features**

The proposed application allows us to create nodes, transreceivers and server. The proposed application avoids intrusion in the network by packet analysis.

#### **4.2.1.3. User characteristics**

Application is developed in such a way that its users are

- Easy to use
- Error free
- Minimal training or no training
- High access control security

#### **4.2.1.4. Assumption & Dependencies**

It is assumed to be ISP providers are cloudlet and allocates the virtual machine requests for the particular cloudlet.

#### **4.2.1.5. Domain Requirements**

This document is the only one that describes the requirements of the system. It is meant for the use by the developers, and will also be the basis for validating the final delivered system. Any changes made to the requirements in the future will have to go through a formal change approval process.

#### **4.2.1.6. User Requirements**

We listed the functional requirements for this application as below.

- ❖ User needs to provide the input requirements such as number of nodes, trans receivers
- ❖ User starts the packet sending process after creating above requirements
- ❖ Random key values for security are generated and allocated accordingly

#### **4.2.2. Non Functional Requirements**

- ❖ Non functional requirements are creating the random values key, node id
- ❖ `rand.nextInt()` is used to create a random values and assigned

#### **4.2.2.1. Product Requirements**

The product requires various configuration creation of ISP networks, user or network admins creates the required data.

#### **4.2.2.1.1. Efficiency**

Performance and resource behavior.

#### **4.2.2.1.2. Reliability**

Maturity, fault tolerance and recoverability.

#### **4.2.2.1.3. Portability**

This software is portable to any location. We can deploy it in server and can access via url from client system. Software easily be transferred to another environment, including install ability.

#### **4.2.2.1.4. Usability**

How easy it is to understand, learn and operate the software system.

#### **4.2.2.2. Organizational Requirements**

Organization which uses the application must provide minimum training requirements to the users who handle the application.

##### **4.2.2.2.1. Implementation Requirements**

Deployment requirements need minimum hardware and software mentioned in the below section. Integrated development environment such as Netbean or eclipse for easy implementation.

##### **4.2.2.2.2. Engineering Standard Requirements**

#### **Hardware Interfaces**

##### **Ethernet**

Ethernet on the AS/400 supports TCP/IP, Advanced Peer-to-Peer Networking (APPN) and advanced program-to-program communications (APPC).

##### **ISDN**

To connect AS/400 to an Integrated Services Digital Network (ISDN) for faster, more accurate data transmission. An ISDN is a public or private digital communications network that can support data, fax,

image, and other services over the same physical interface. can use other protocols on ISDN, such as IDLC and X.25.

#### **4.2.2.3. Operational Requirements**

##### **User Interfaces**

User Interfaces are Graphical User Interfaces in this product. Users are communicated with Buttons to clear the content or send data to the destination. User can enter the data through the textbox. User can interact with text area to enter the multiple line of text.

##### **Software Interfaces**

This software is interacted with the TCP/IP protocol. This product is interacted with the Socket and listening on unused ports. This product is interacted with the Server Socket and listening on unused ports.

##### **Communications Interfaces**

The TCP/IP protocol will be used to facilitate communications between the client and server.

##### **Performance Requirements**

The maximum satisfactory response time to be experienced most of the time for each distinct type of user-computer interaction, along with a definition of most of the time. Response time is measured from the time that the user performs the action that says "Go" until the user receives enough feedback from the computer to continue the task. It is the user's subjective wait time. It is not from entry to a subroutine until the first write statement. If the user denies interest in response time and indicates that only the result is of interest, you can ask whether "ten times your current estimate of stand-alone execution time" would be acceptable. If the answer is "yes," you can proceed to discuss throughput.

##### **Safety Requirements**

The software may be safety-critical. If so, there are issues associated with its integrity level. The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level. There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable. If a computer

system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level. Systems with different requirements for safety levels must be separated. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

### **Security Requirements**

Do not block the some available ports through the windows firewall. Two machines should be connected with LAN setting.

#### **4.2.3. System Requirements**

Are the required functions available, including interoperability and security.

#### **4.2.3. SYSTEM REQUIREMENTS**

##### **HARDWARE REQUIREMENTS:**

Processor	: Any Processor above 500 MHz.
Ram	: 128Mb.
Hard Disk	: 10 Gb.
Compact Disk	: 650 Mb.
Input device	: Standard Keyboard and Mouse.
Output device	: VGA and High Resolution Monitor.

##### **SOFTWARE REQUIREMENTS:**

Operating System	: Windows Family.
Language	: JDK 1.5
Front End	: Java Swing

##### **SOFTWARE DESCRIPTION:**

## **JAVA**

Java is an object-oriented multithread programming languages .It is designed to be small, simple and portable across different platforms as well as operating systems.

### **FEATURES OF JAVA**

#### **Platform Independence**

- The *Write-Once-Run-Anywhere* ideal has not been achieved (tuning for different platforms usually required), but closer than with other languages.

#### **Object Oriented**

- Object oriented throughout - no coding outside of class definitions, including main().
- An extensive class library available in the core language packages.

#### **Compiler/Interpreter Combo**

- Code is compiled to byte codes that are interpreted by a Java virtual machines (JVM).
- This provides portability to any machine for which a virtual machine has been written.
- The two steps of compilation and interpretation allow for extensive code checking and improved security.

#### **Robust**

- Exception handling built-in, strong type checking (that is, all data must be declared an explicit type), local variables must be initialized.

#### **Several features of C & C++ eliminated:**

- No memory pointers
- No preprocessor
- Array index limit checking



## Automatic Memory Management

- Automatic garbage collection - memory management handled by JVM.

## Security

- No memory pointers
- Programs run inside the virtual machine sandbox.
- Array index limit checking
- Code pathologies reduced by
  - *Byte code verifier* - checks classes after loading
  - *Class loader* - confines objects to unique namespaces. Prevents loading a hacked "java.lang.SecurityManager" class, for example.
  - *Security manager* - determines what resources a class can access such as reading and writing to the local disk.

## Dynamic Binding

- The linking of data and methods to where they are located is done at run-time.
- New classes can be loaded while a program is running. Linking is done *on the fly*.
- Even if libraries are recompiled, there is no need to recompile code that uses classes in those libraries.
- This differs from C++, which uses static binding. This can result in *fragile* classes for cases where linked code is changed and memory pointers then point to the wrong addresses.

## Good Performance

- Interpretation of byte codes slowed performance in early versions, but advanced virtual machines with adaptive and just-in-time compilation and other techniques now typically provide performance up to 50% to 100% the speed of C++ programs.

## Threading

- *Lightweight* processes, called threads, can easily be spun off to perform multiprocessing.
- Can take advantage of multiprocessors where available

- Great for multimedia displays.

### **Built-in Networking**

- Java was designed with networking in mind and comes with many classes to develop sophisticated Internet communications.

IMP applications are called IMlets, but in reality they are MIDlets. They subclass MIDlet, and follow the same packaging, deployment, security and life-cycle as MIDlets.

### **Connected Device Configuration**

CDC is a smaller subset of Java SE, containing almost all the libraries that are not GUI related.

### **Foundation Profile**

A headless version of Java SE.

### **Personal Basis Profile**

Extends the Foundation Profile to include lightweight GUI support in the form of an AWT subset.

### **Personal Profile**

This extension of Personal Basis Profile includes a more comprehensive AWT subset and adds applet support.

## Net Beans

Net Beans A Java-based development environment (IDE) and platform originally developed by Sun. It includes user interface functions, source code editor, GUI editor, version control as well as support for distributed applications (CORBA, RMI, etc.) and Web applications (JSPs, servlets, etc.).

In 1999, Sun acquired NetBeans Developer from NetBeans and rebranded it as Forte for Java Community Edition (Sun acquired Forte in 1999). In 2000, Sun made the NetBeans IDE open source.

1. **GUI:** The major requirement of today's developers is to have a good User Interface for their users. They can provide whatever functionality they need but it's the GUI that lets the user better know the existence of that particular functionality and its easier for them to click and select than type something on a black boring screen. Thus, today's developers need IDE's such as netbeans that develop ready made windows forms with all the required buttons, labels, text boxes and like that can be tailor made for the program in question.

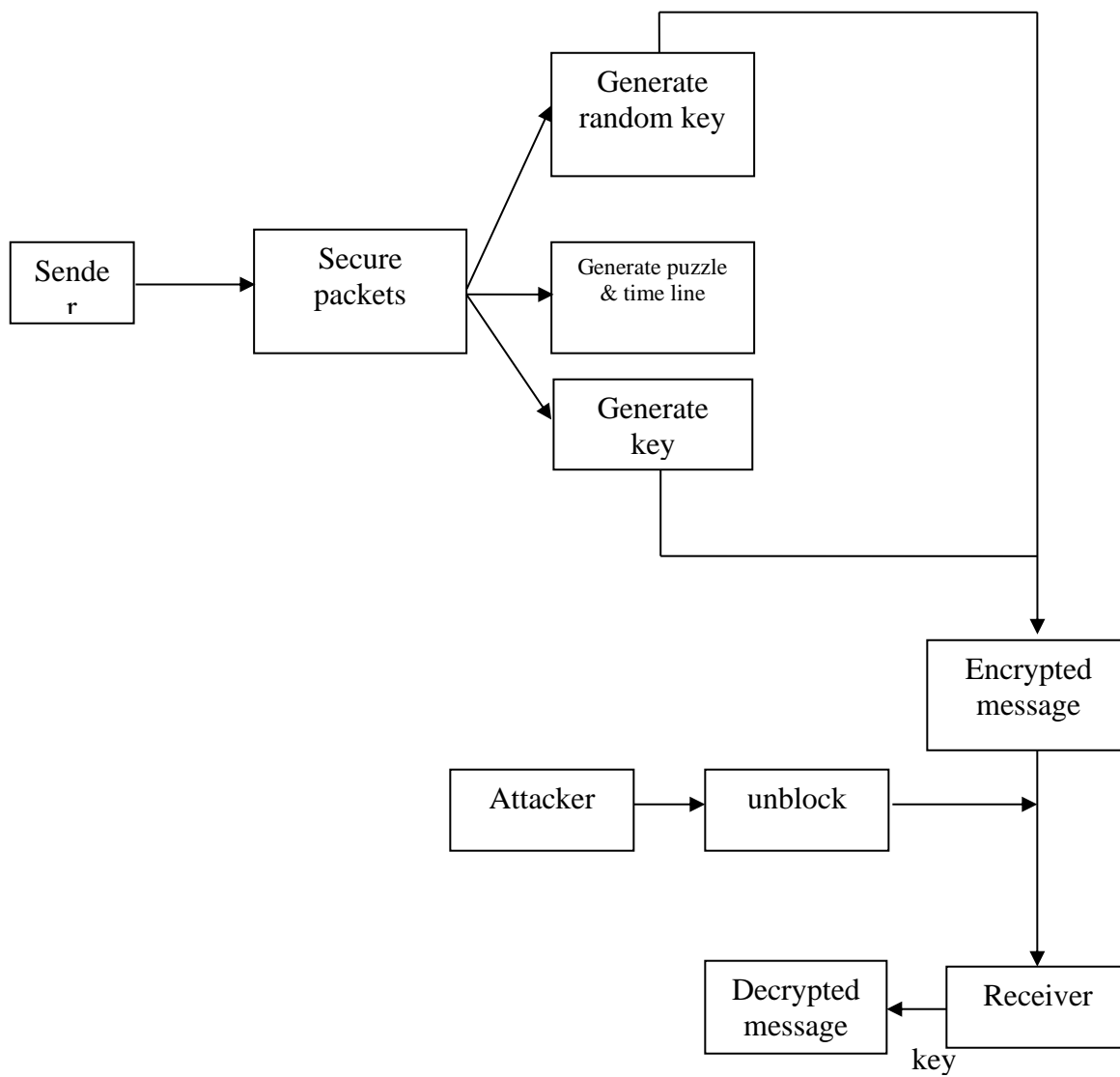
## CHAPTER 4

### DESIGN

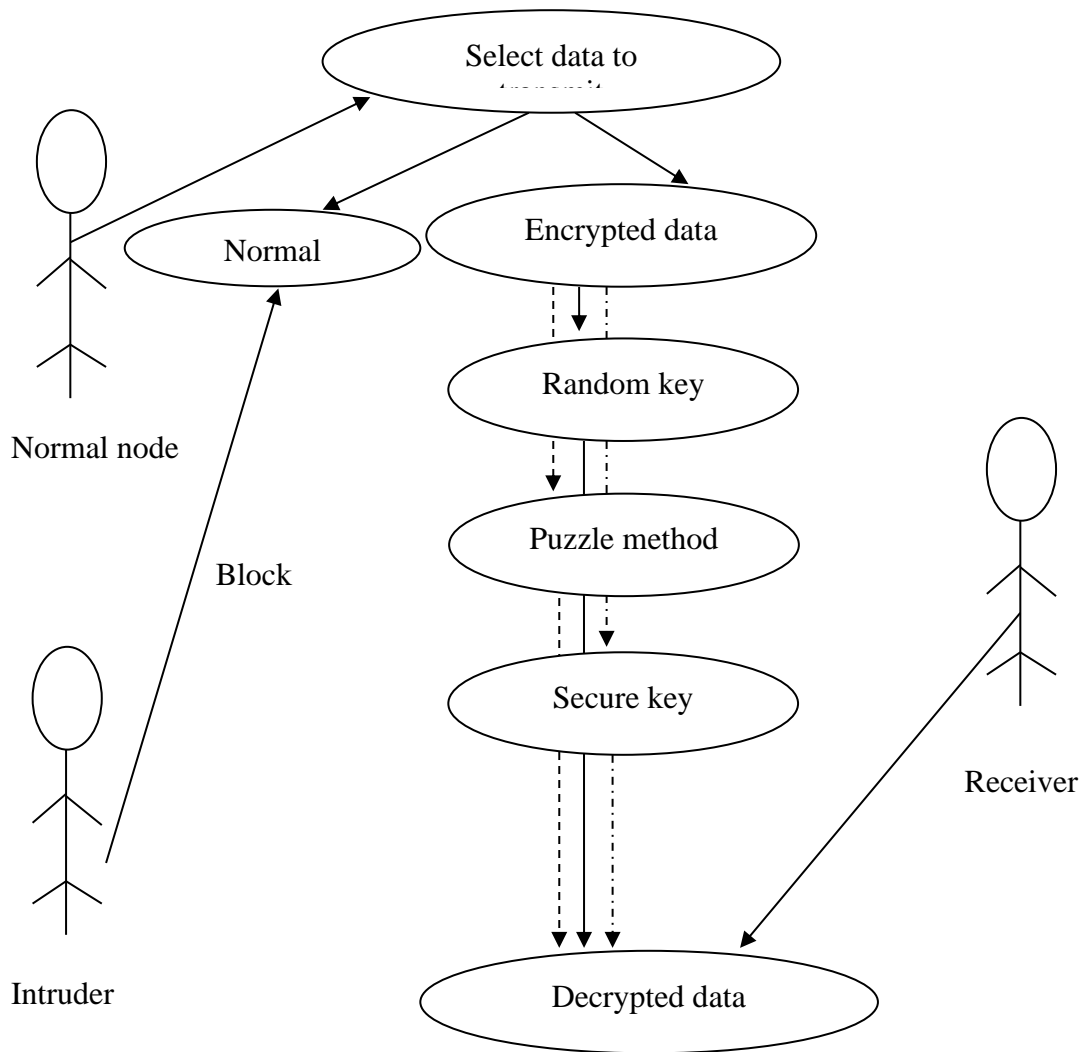
#### SYSTEM DESIGN

This chapter gives overview of architecture design, dataset for implementation, algorithm used and UML designs.

#### SYSTEM ARCHITECTURE

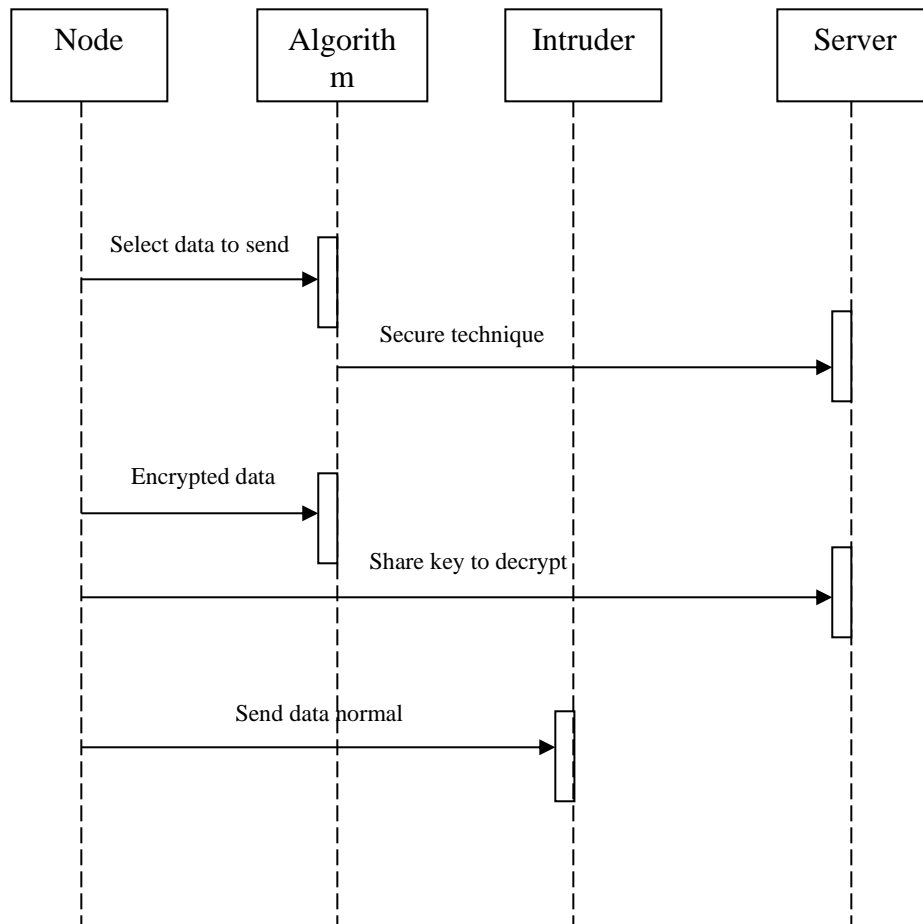


## USE CASE DIAGRAM



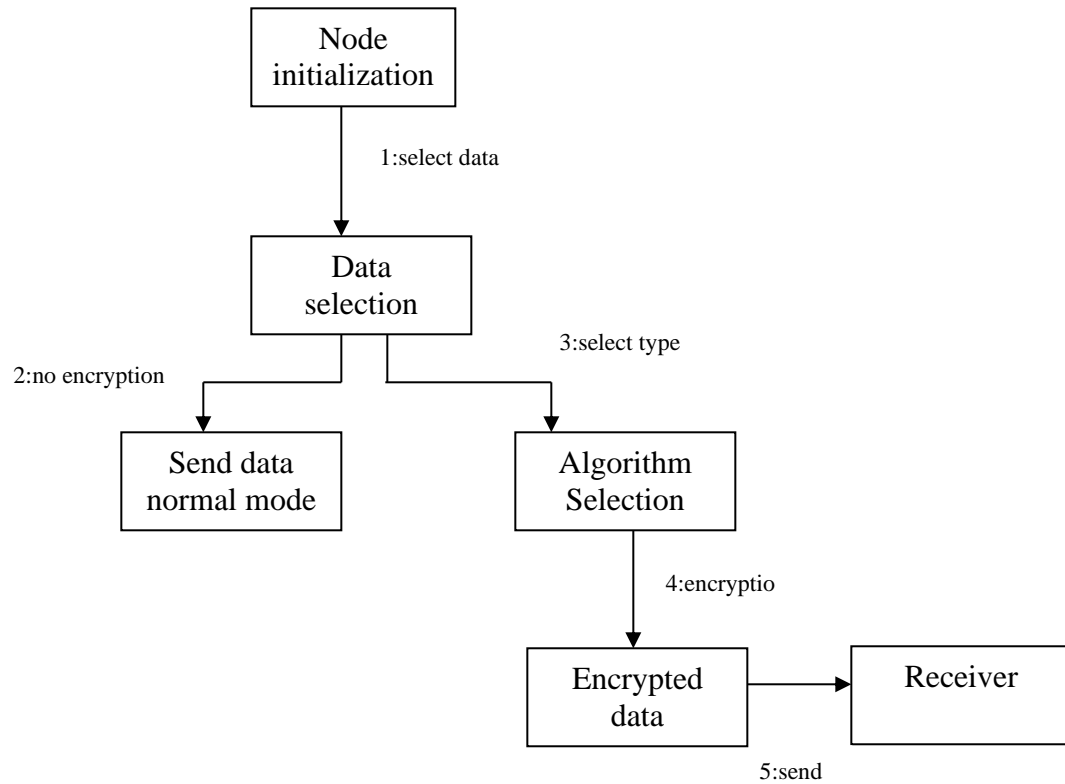
Normal node chooses data to send and the data is encrypted using any one of the secure methods and send to receiver, sender shares the decrypt key to decrypt the data. Intruder is considered as one of the other user, who tries blocking the data content while sending normally.

## SEQUENCE DIAGRAM



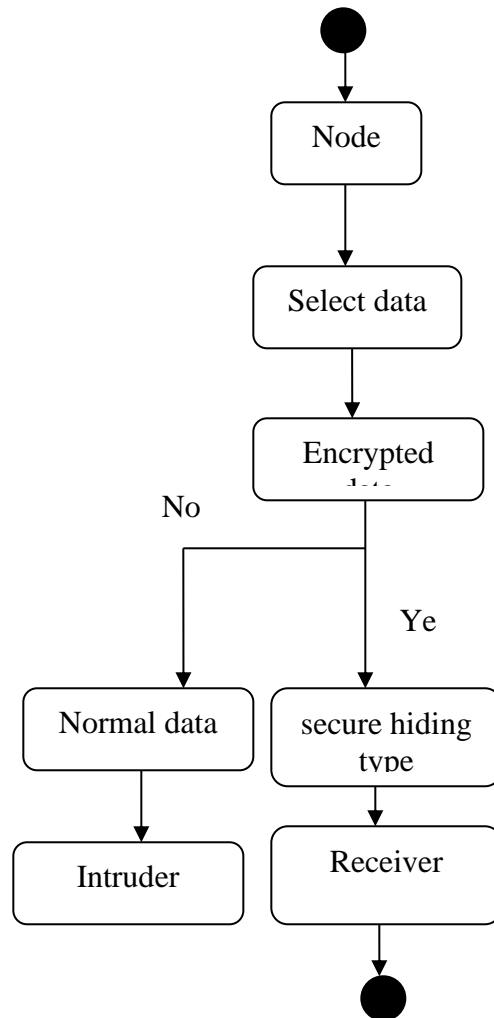
Node are initialized and chooses data to send and the data is encrypted using one of the secure methods and send to receiver, sender shares the decrypt key to decrypt the data. Intruder is considered as one of the other user, who tries blocking the data content while sending normally.

## COLALBORATION DIAGRAM



First step that nodes are initialized and data selected to send to the receiver is the second step. Selecting the hiding technique is the third step and then encrypted data send to server is the fifth step.

## ACTIVITY DIAGRAM



When the node selects data to send, the data is encrypted via any of the hiding techniques. When the data is sent via normal mode without using any of the hiding technique, then it can be blocked by intruder.



## CLASS DIAGRAM



**Fig: Class Diagram**

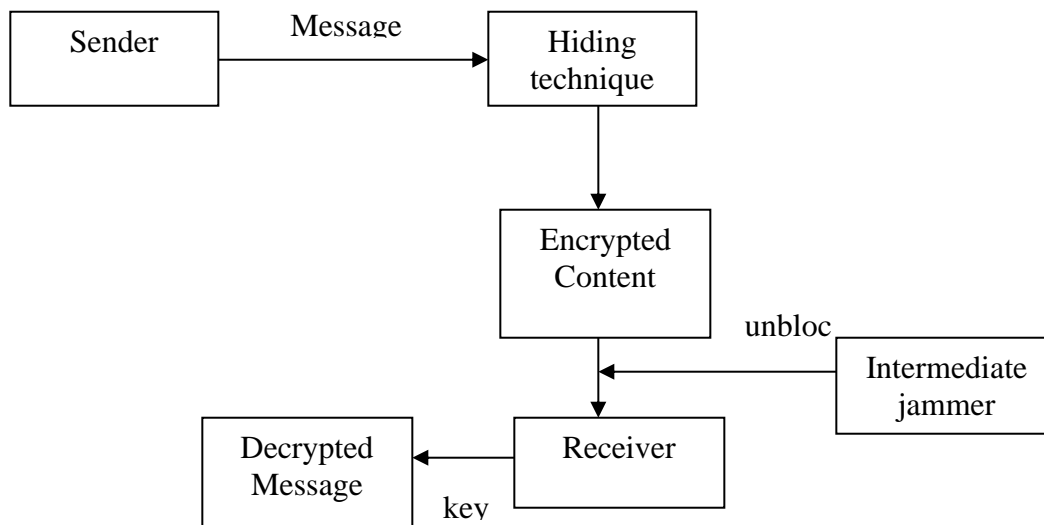
## CHAPTER 5

### IMPLEMENTATION

The proposed work is implemented in JDK 1.8 using SWING package in java. There are three entities designed here are nodes, intruder, receiver. The socket connection is done between nodes and receiver/server. The localhost address or the IP address of the sender and receiver are considered for the connection establishment.

### REAL TIME PACKET CLASSIFICATION

At the Physical layer, a packet  $m$  is encoded, interleaved, and modulated before it is transmitted over the network channel. At the receiver, the signal is demodulated, deinterleaved and decoded to recover the original packet  $m$ . Nodes A and B communicate via a wireless link. Within the communication range of both A and B there is a intruder node J. When A transmits a packet  $m$  to B, node J classifies  $m$  by receiving only the first few bytes of  $m$ . J then corrupts  $m$  beyond recovery by interfering with its reception at B.

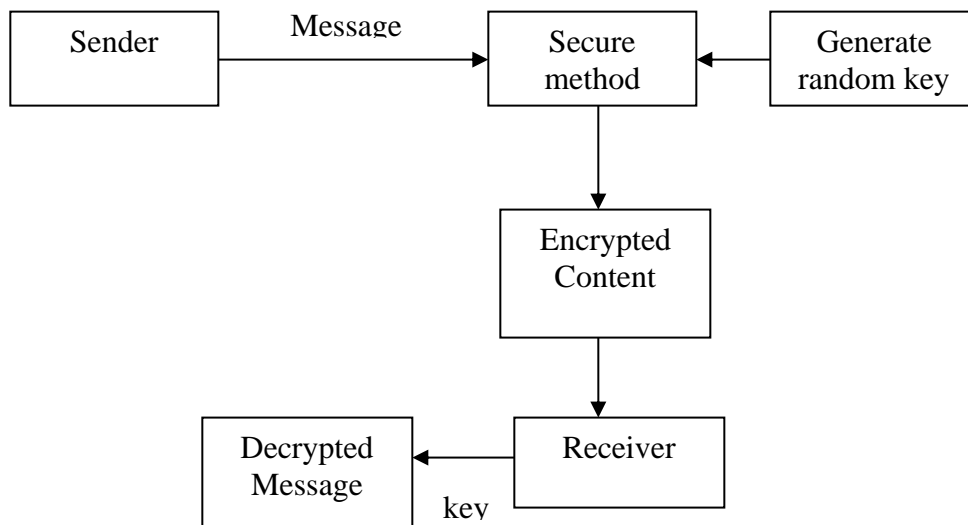


### ADVERSARY DESIGN

Adversary is in control of the communication medium and can intrude messages at any part of the network of his choosing. The adversary can operate in full-duplex mode, thus being able to receive and transmit simultaneously. This can be achieved, for example, with the use of multi-radio transceivers. It is assumed that the adversary can pro-actively intrude a number of bits just below the security capability early in the transmission. When the adversary is introduced, the data packets from the node cannot be reached at receiver.

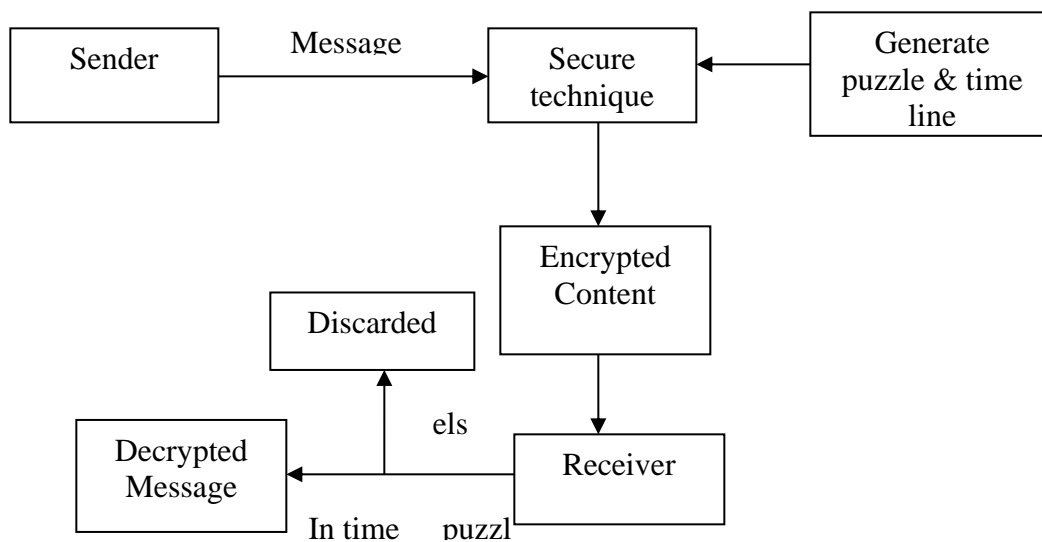
## PROBE ATTACK PREVENTION

The probe attacks are aimed at acquiring information about the target network from a source that is often external to the network. Hence, basic connection level features such as the “duration of connection” and “source bytes” are significant while features like “number of files creations” and “number of files accessed” are not expected to provide information for detecting probes. Random key generation and encryption is proposed for preventing this attack, which is based on symmetric cryptography. Assume that the sender has a packet for Receiver. First Sender sends a message with symmetric encryption algorithm with a randomly selected key of some desired key length  $s$ . The role of the verifier is assumed by any receiver, which receives a packets. To transmit message, the sender computes the corresponding key and then broadcasts. The receiver upon giving the key can able to decode and receive packets.



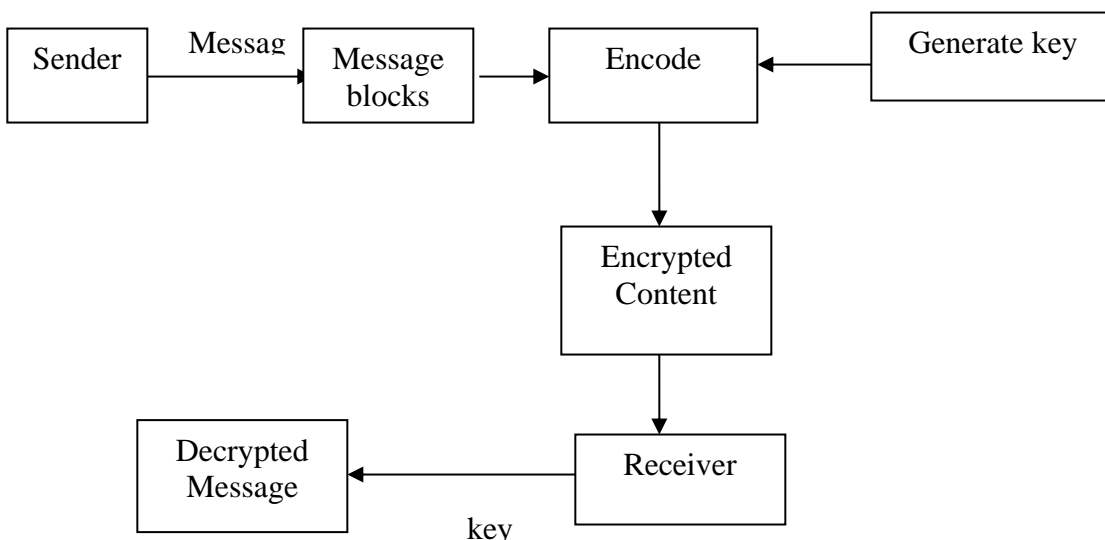
## DDOS ATTACK PREVENTION

The DoS attacks are meant to force the target to stop the services, which are provided by flooding it with illegitimate requests. Traffic features such as the “percentage of connections having same destination host and same service” and packet level features such as the “source bytes” and “percentage of packets with errors” are significant. A sender have a packet for transmission. The sender selects a random key  $k$ , of a desired length. The sender generates a puzzle along with key time. After generating the puzzle, the sender broadcasts message. The receiver node, solves the received puzzle to recover key. The key has to given properly to get and decode the message at receiver node.



## ROOT AND LOCAL ATTACK PREVENTION

The R2L attacks are one of the most difficult to detect as they involve the network level and the host level features. Network level features are selected such as the “duration of connection” and “service requested” for detecting R2L attacks. The packets are pre-processed by secure transmission method before transmission but remain unencrypted. The intruder cannot perform packet classification until all pseudo-messages corresponding to the original packet have been received. The sender send the message along with security key after encoding the message. The receiver receives the message with key in the given time seconds, so can decode and receive message. If the receiver not given key in the given time seconds, it cannot able to receive message.



## **CHAPTER 6**

### **TESTING**

#### **Introduction:**

After finishing the development of any computer based system the next complicated time consuming process is system testing. During the time of testing only the development company can know that, how far the user requirements have been met out, and so on.

Software testing is an important element of the software quality assurance and represents the ultimate review of specification, design and coding. The increasing feasibility of software as a system and the cost associated with the software failures are motivated forces for well planned through testing.

#### **Testing Objectives**

These are several rules that can save as testing objectives they are:

- Testing is a process of executing program with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.

Following are the some of the testing methods applied to this effective project:

#### **SOURCE CODE TESTING:**

This examines the logic of the system. If we are getting the output that is required by the user, then we can say that the logic is perfect.

#### **SPECIFICATION TESTING:**

We can set with, what program should do and how it should perform under various condition. This testing is a comparative study of evolution of system performance and system requirements.

#### **MODULE LEVEL TESTING:**

In this the error will be found at each individual module, it encourages the programmer to find and rectify the errors without affecting the other modules.

#### **UNIT TESTING:**

Unit testing focuses on verifying the effort on the smallest unit of software-module. The local data structure is examined to ensure that the data stored temporarily maintains its integrity during all steps in the algorithm's execution. Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.

#### **INTEGRATION TESTING:**

Data can be tested across an interface. One module can have an inadvertent, adverse effect on the other. **Integration testing** is a systematic technique for constructing a program structure while conducting tests to uncover errors associated with interring.

#### **VALIDATION TESTING:**

It begins after the integration testing is successfully assembled. Validation succeeds when the software functions in a manner that can be reasonably accepted by the client. In this the majority of the validation is done during the data entry operation where there is a maximum possibility of entering wrong data. Other validation will be performed in all process where correct details and data should be entered to get the required results.

#### **RECOVERY TESTING:**

**Recovery Testing** is a system that forces the software to fail in variety of ways and verifies that the recovery is properly performed. If recovery is automatic, re-initialization, and data recovery are each evaluated for correctness.

#### **SECURITY TESTING:**

Security testing attempts to verify that protection mechanism built into system will in fact protect it from improper penetration. The tester may attempt to acquire password through external clerical means, may attack the system with custom software design to break down any defenses to others, and may purposely cause errors.

#### **PERFORMANCE TESTING:**

Performance Testing is used to test runtime performance of software within the context of an integrated system. Performance test are often coupled with stress testing and require both software instrumentation.



## **BLACKBOX TESTING:**

**Black- box testing** focuses on functional requirement of software. It enables to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing attempts to find error in the following category:

- ❖ Incorrect or missing function
- ❖ Interface errors
- ❖ Errors in data structures or external database access and performance errors.

## **OUTPUT TESTING:**

After performing the validation testing, the next step is output testing of the proposed system since no system would be termed as useful until it does produce the required output in the specified format. **Output format** is considered in two ways, the **screen format** and the **printer format**.

## **USER ACCEPTANCE TESTING:**

User Acceptance Testing is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system users at the time of developing and making changes whenever required.

## TEST CASES

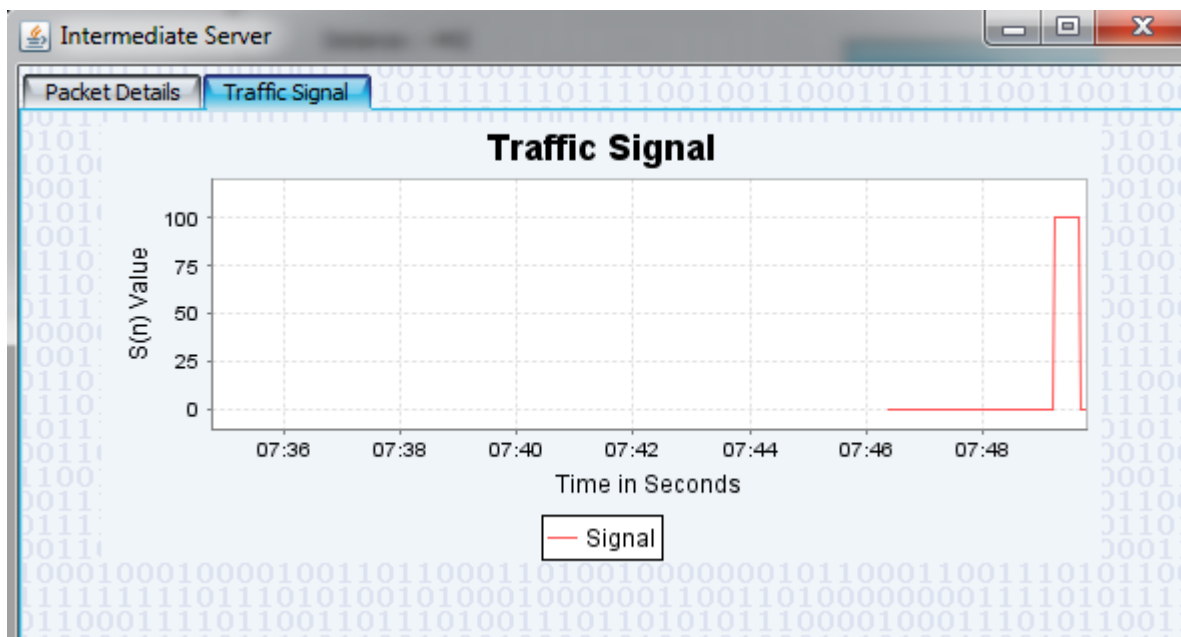
Sl. No	Test Case Name	Test Procedure	Pre-Condition	Expected Result	Passed/ failed
1	Client Validation	Click on Send button without choosing any value	Don't perform Leader Election and don't choose any algorithm	Alter "Please Select All the Fields"	Passed
2	Client Validation	Click on Send button without choosing Leader Election	Don't perform Leader Election and choose any one of the algorithm	Alter "Please Select All the Fields"	Passed
3	Client Validation	Click on Send button after choosing Cryptographic puzzles algorithm	Perform Leader Election and choose Cryptographic puzzles algorithm, enter time line 3 sec. then send after waiting for 5 sec	Alter "Time Line Expired"	Passed
4	Client Validation	Click on Send button after choosing Cryptographic puzzles algorithm	Perform Leader Election and choose Cryptographic puzzles algorithm, enter time line 3 sec. then send after waiting for 5 sec	Message should not reach server	Passed
5	Client Validation	Click on Send button after choosing Cryptographic	Perform Leader Election and choose Cryptographic puzzles algorithm, enter	Incorrect solution	Passed

		puzzles algorithm	time line 30 sec. then send by giving wrong puzzle value		
--	--	----------------------	---	--	--

## CHAPTER 7

### RESULT AND ANALYSIS

Implemented five machine learning algorithms on the given dataset for phishing website detection shows that Logistics regression model outperforms other models. The accuracy of Random forest is high compared to other machine learning algorithms. We study preventive Jamming attacks under four special cases such as All Or Nothing Transformations, Cryptographic Puzzles, Strong Hiding Commitment Schemes and as Normal Mode. In this work, one server, one intermediate server, which is assumed to be the attacker under two cases block and unblock.



## **CHAPTER 8**

### **CONCLUSION AND FUTURE SCOPE**

Intrusion attacks in wireless networks have been studied. We considered an internal adversary model in which the intruder is part of the network under attack, thus being aware of the protocol specifications and shared network secrets. We evaluated the impact of attacks on network protocols such as TCP and routing. Three schemes have been proposed that transform packets secure to the network by preventing real-time packet classification. The schemes combine cryptographic primitives, cryptographic puzzles, and timely transmission with physical layer characteristics.

Future work may be extended to evaluate the performance of hybrid solutions, by combining the three approaches proposed in this paper. The impact of interference between nodes that belong to two or more jamming regions (and its dependency on the node density) is also a subject for future work.

## REFERENCES

- [1] Mo Kun. Network Intrusion Detection System Based on LightGBM [J]. Information Security Research, 2019, No. 2, 152-156.
- [2] Zhang Chuangji. Network Intrusion Detection Algorithm Based on Symbol Envelope Amplitude Extraction[J]. "Intelligent Computers and Applications", No. 2, 2019, 155-158.
- [3] Long Rui. Discussion on computer network intrusion detection system matching algorithm [J]. Computer Knowledge and Technology, No. 4, 2019, 35-38.
- [4] Ma Wei. Research on the performance of network intrusion detection system [J]. Journal of Chifeng College: Natural Science Edition, No. 11 of 2018, 48-51.
- [5] Wang Shuguang. Deployment and testing of network intrusion detection system [J]. Electronic World, No. 11 of 2018, 31-32.
- [6] Qian Tieyun. Intrusion detection method based on deep neural network[J]. Journal of Huazhong University of Science and Technology(Natural Science Edition) 2018, Vol.46, No.1 P6-10 1671-4512.
- [7] Jiang Bin. Network Intrusion Detection Model Based on Feature Selection[J]. Modern Electronic Technology 2019, Vol.42, No.1 P87 90,94 1004-373X.
- [8] Liang Chuan. Immune theory of ship network intrusion detection method [J]. Ship Science and Technology 2019 No. 12 P160-162 1672-7649.
- [9] Jia Fan. Intrusion Detection Algorithm Based on Convolutional Neural Network[J]. Journal of Beijing Institute of Technology 2017 No.12 P1271-1275 1001-0645.

## ANNEXURE

### SOURCE CODE

#### ServerDesignForm.java

```
package com.mycompany.design;

import java.awt.Color;
import org.jvnet.substance.SubstanceLookAndFeel;
import java.awt.Dimension;
import java.awt.Toolkit;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JSplitPane;
import javax.swing.JTable;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.UIManager;
import javax.swing.table.DefaultTableModel;

import org.jvnet.substance.SubstanceLookAndFeel;

import com.mycompany.logic.ServerLogic;
import com.mycompany.support.SystemProperties;

public class ServerDesignForm extends JFrame {
    public JLabel expressionLabel;
    private ServerLogic serverLogic;
    public JTextArea jtextArea;
    public static DefaultTableModel model;
    public static JTable table;
    public static JScrollPane jScrollPane ;
    private SystemProperties systemProperties = new SystemProperties();
    private String[] initValues = systemProperties.ServerValues();

    public static DefaultTableModel model1 ;
    public static JTable table1 ;
    public static JScrollPane jScrollPane1 ;
    public JPanel jp4;
    public JPanel jp5;
    public JTextField expressionText;
    public JTextField expressionValue;

    static
    {
        try
        {
            SubstanceLookAndFeel

            .setCurrentWatermark("org.jvnet.substance.watermark.SubstanceBinaryWatermark");
            SubstanceLookAndFeel
```

```

        .setCurrentTheme("org.jvnet.substance.theme.SubstanceInvertedTheme");
        SubstanceLookAndFeel

    .setCurrentGradientPainter("org.jvnet.substance.painter.SpecularGradientPainter"
);
    SubstanceLookAndFeel

    .setCurrentButtonShaper("org.jvnet.substance.button.ClassicButtonShaper");
    UIManager.setLookAndFeel(new SubstanceLookAndFeel());
    }
    catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    }
}
public void Design() {
    try {
        model = new DefaultTableModel();
        table = new JTable(model);
        jScrollPane = new JScrollPane(table);

        model1 = new DefaultTableModel();
        table1 = new JTable(model1);
        jScrollPane1 = new JScrollPane(table1);

        serverLogic = new ServerLogic(Integer.parseInt(initValues[5]),
"whois", this, "initValues[2]",
        "initValues[4]", "nodeName");
        setTitle("Server");
        JPanel jp1 = new JPanel();
        JPanel jp2 = new JPanel();
        // JPanel jp3 = new JPanel();
        jp1.setLayout(null);
        jp2.setLayout(null);

        JPanel jp3 = Expression();
        // jp1.setBackground(Color.GRAY);
        JLabel jLabel = new JLabel("Received Message :");
        jLabel.setBounds(30, 10, 150, 50);
        jp1.add(jLabel);
        jp3.setBounds(0, 350, 300, 300);
        // jp3.setBackground(Color.BLUE);
        // jp3.add(jLabel);
        jp1.add(jp3);
        JLabel jLabel1 = new JLabel("Technical Details :");
        jLabel1.setBounds(30, 10, 150, 50);
        jp2.add(jLabel1);

        jp4 = NormalDesign();
        jp5 = AlgorithmDesign();
        //jp4.setBackground(Color.BLUE);

        jp5.setLayout(null);
        jp5.setBounds(50, 60, 900, 300);
        jp2.add(jp5);

        jp5.setVisible(false);

```



```

        jp4.setLayout(null);
        jp4.setBounds(50, 60, 500, 300);
        jp2.add(jp4);

        jp4.setVisible(false);
        // jp1.setSize(100, 100);
        jTextArea = new JTextArea();
        jTextArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(jTextArea,
            JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
            JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
        scrollPane.setBounds(20, 60, 300, 240);
        jp1.add(scrollPane);

        JSplitPane splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
            false, jp1, jp2);
        splitPane.setOneTouchExpandable(true);
        splitPane.setDividerLocation(350);
        add(splitPane);

        setSize(1200, 700);
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        Dimension size = getSize();
        screenSize.height = screenSize.height / 2;
        screenSize.width = screenSize.width / 2;
        size.height = size.height / 2;
        size.width = size.width / 2;
        int y = screenSize.height - size.height;
        int x = screenSize.width - size.width;
        setLocation(x, y);

        setVisible(true);
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    }
}

public JPanel Expression() {
    JPanel jp3 = new JPanel();
    try {
        jp3.setLayout(null);
        JLabel jLabel = new JLabel("Puzzle value's : ");
        jLabel.setBounds(20, 20, 200, 50);
        jp3.add(jLabel);
        JLabel jLabel1 = new JLabel("Generated value : ");
        jLabel1.setBounds(20, 80, 200, 50);
        jp3.add(jLabel1);
        expressionLabel = new JLabel();
        expressionLabel.setBounds(170, 30, 100, 30);
        jp3.add(expressionLabel);
        expressionText = new JTextField();
        expressionText.setBounds(170, 90, 100, 30);
        expressionText.setEditable(false);
        jp3.add(expressionText);
        JLabel jLabelResult = new JLabel("Generated Result : ");
        jLabelResult.setBounds(20, 130, 200, 50);
    }
}

```

```

        jp3.add(jLabelResult);
        expressionValue = new JTextField();
        expressionValue.setBounds(170, 140, 100, 30);
        jp3.add(expressionValue);
        expressionValue.setEditable(false);
    } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
    }
    return jp3;
}

public JPanel AlgorithmDesign() {
    JPanel jp4 = new JPanel();
    try {

        jp4.setLayout(null);
        model.addColumn("Algorithm Type");
        model.addColumn("Secret key");
        model.addColumn("No of Packets");
        model.addColumn("Secret keypath");
        model.addColumn("Issued Time");
        model.addColumn("Resolved Time");
        model.addColumn("Packet's order");
        jScrollPane.setBounds(0, 50, 700, 250);
        jp4.add(jScrollPane);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return jp4;
}

public JPanel NormalDesign() {
    JPanel jp4 = new JPanel();
    try {

        jp4.setLayout(null);
        model1.addColumn("Blocked IP");
        model1.addColumn("No of Packets");
        model1.addColumn("Blocked Packets");
        jScrollPane1.setBounds(0, 50, 400, 250);
        jp4.add(jScrollPane1);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return jp4;
}

public static void main(String[] args) {
    ServerDesignForm serverDesignForm = new ServerDesignForm();
    serverDesignForm.Design();
}
}

```

## Receiver.java

```
package com.mycompany.logic;
```

```

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.net.MulticastSocket;
import java.net.Socket;
import java.util.Date;
import java.util.Iterator;
import java.util.Properties;
import java.util.Random;
import java.util.StringTokenizer;
import java.util.TreeMap;
import java.util.TreeSet;
import java.util.Vector;

import com.mycompany.design.NodeDesignForm;
import com.mycompany.support.SocketConnection;
import com.mycompany.support.SplitDetails;

public class Receiver extends Thread {

    public static String serverCache;

    public static TreeMap<Integer, String> transValue = new TreeMap<Integer,
String>();

    public TreeSet<Integer> treeValue = new TreeSet<Integer>();

    int size = 0;

    SocketConnection socketConnection = new SocketConnection();

    SplitDetails splitDetails = new SplitDetails();

    MulticastSocket ms;

    String localhost = "";

    // MultiCastString mstr;

    String MuCst = "", s = null, cur_node;

    String dist;

    Vector myNegh;

    Vector vec = new Vector();

    Vector prop_con = new Vector();

    Vector ack_send = new Vector();

    public TreeSet tree = new TreeSet();

    public static TreeMap<String, String> curd = new TreeMap<String, String>();

    public TreeSet all_nodes = new TreeSet();

```

```

public TreeMap clusterHead = new TreeMap();

StringTokenizer str_port;

String nodname;

String port;

Iterator ii;

String sys_port = "", sys_name = "";

Socket soc;

ObjectOutputStream oo;

public TreeMap send_time = new TreeMap();

Date dd;

public long tim;

String nodd;

int i = 0;

FileOutputStream fo;

Properties prop;

FileInputStream fin;

String data = "";

boolean flag = false;

public int len;

public TreeSet<String> nei_tree = new TreeSet<String>();

String region;

String who;
NodeDesignForm nodeDesignForm;

public TreeMap<String, Integer> mobilityName = new TreeMap<String, Integer>();
public TreeMap<Integer, String> mobilityMap = new TreeMap<Integer, String>();

public Receiver(String port, String name, String region,
                NodeDesignForm nodeDesignForm, String dis) {
    this.nodeDesignForm = nodeDesignForm;
    this.port = port;
    this.nodd = name;
    this.region = region;
    this.dist = dis;
    // new stop();
    start();
}

```

```

    }

    public void setWho(String whosis) {
        this.who = whosis;
    }

    public void setname(String nod) {
        // System.out.println("cur_node..." + nod);
        cur_node = nod;
    }

    /*
    * public void setMul(String dis) { dist = dis; }
    */

    public void run() {
        try {
            while (true) {
                ms = new MulticastSocket(5454);
                InetAddress ia = InetAddress.getByName("228.2.5.11");
                ms.joinGroup(ia);
                byte[] b = new byte[1000];
                DatagramPacket dp = new DatagramPacket(b, b.length);
                ms.receive(dp);
                s = new String(dp.getData()).trim();
                StringTokenizer str = new StringTokenizer(s.trim(), ":");
                // String whosis = str.nextToken(":");
                Random rand = new Random();
                int value = rand.nextInt(10000);
                // System.out.println("SSSSS " + s);

                String reg = str.nextToken(":");
                String dis = str.nextToken(":");
                String nodeName = str.nextToken(":");
                String port = str.nextToken(":");
                String sysName = str.nextToken(":");
                String mobilityScore = str.nextToken(":");
                curd.put(nodeName, port + ":" + sysName);
                //System.out.println(s);
                if (mobilityName.containsKey(nodeName)) {
                    int mobilityValue = mobilityName.get(nodeName);
                    mobilityMap.remove(mobilityValue);
                    mobilityMap.put(Integer.parseInt(mobilityScore),
mobilityName.put(nodeName,
Integer.parseInt(mobilityScore)));
                } else {
                    mobilityMap.put(Integer.parseInt(mobilityScore),
mobilityName.put(nodeName,
Integer.parseInt(mobilityScore)));
                }

                /*
                * System.out.println("nodeName :" + nodeName);
                * System.out.println("region :" + region);
                * System.out.println("reg :" + reg);
                */
            }
        }
    }

```

```

        if (!nodd.equalsIgnoreCase(nodeName)) {
            if (region.equals("region1")) {
                /*
                 * System.out.println("Node 1 :" + nodeName);
                 * System.out.println("cat :" + cat);
                 * System.out.println("category :" + category);
                 */

                if (!reg.equals("region1") &&
reg.equals("region2")) {
                    if (Integer.parseInt(dis) <=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
                if (reg.equals("region1")) {
                    if (Integer.parseInt(dis) !=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
            }
            if (region.equals("region2")) {
                if (!reg.equals("region2") &&
reg.equals("region1")) {
                    if (Integer.parseInt(dis) >=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
                if (!reg.equals("region2") &&
reg.equals("region3")) {
                    if (Integer.parseInt(dis) <=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
                if (reg.equals("region2")) {
                    if (Integer.parseInt(dis) !=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
            }
            if (region.equals("region3")) {
                if (!reg.equals("region3") &&
reg.equals("region2")) {
                    if (Integer.parseInt(dis) >=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
                if (reg.equals("region3")) {
                    if (Integer.parseInt(dis) !=
Integer.parseInt(dist)) {
                        nei_tree.add(nodeName);
                    }
                }
            }
        }
    }
}

```

```

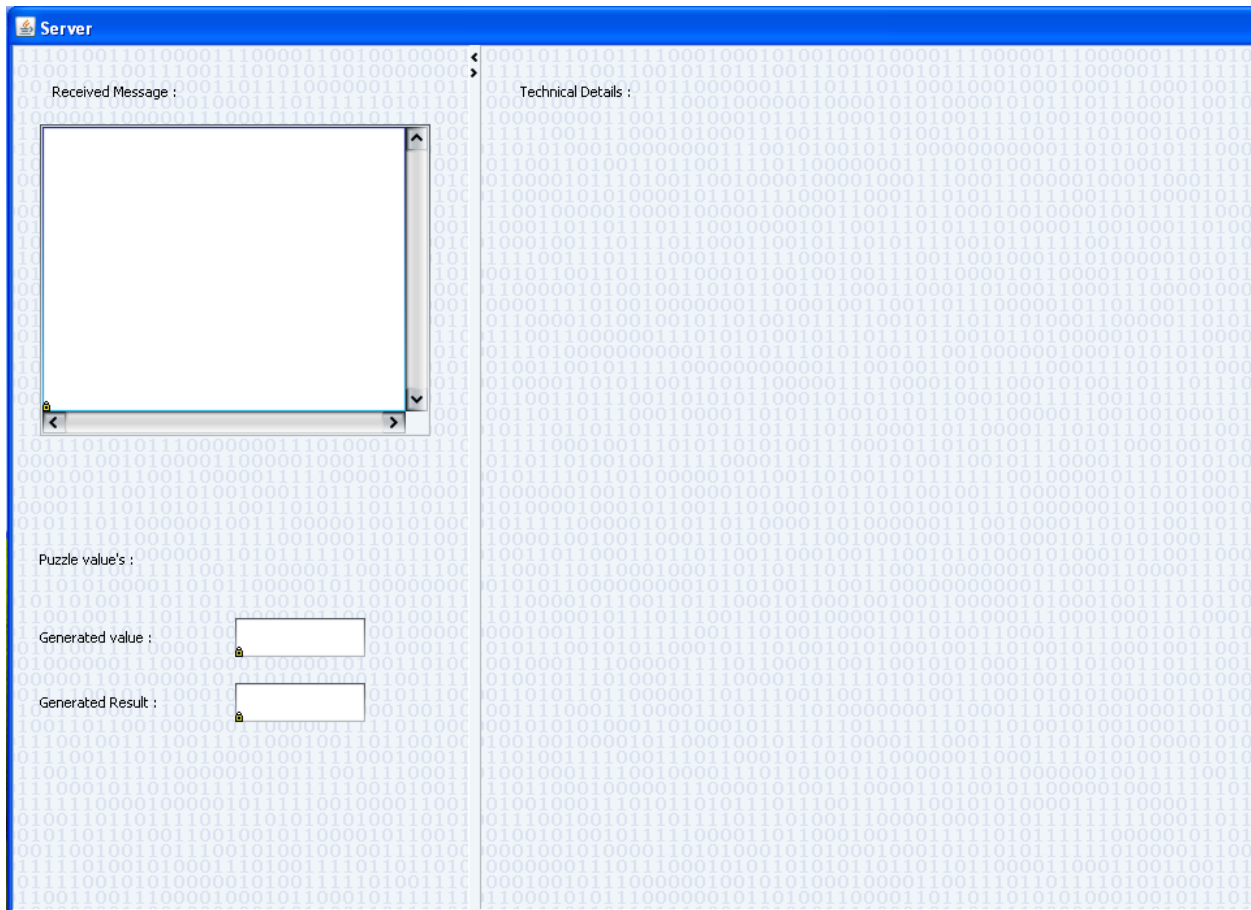
        }
    }
    /*
    * System.out.println("Node "+nodd);
    * System.out.println("Neighbour "+nei_tree);
    */
    Iterator iter = nei_tree.iterator();
    String text = "";
    while (iter.hasNext()) {
        String temp = iter.next().toString();
        if (!temp.equals(nodd))
            text += temp + " \n";
    }
    nodeDesignForm.jTextAreaNeigh.setText(text);
}

}
} catch (Exception ee) {
    ee.printStackTrace();
}
}
}

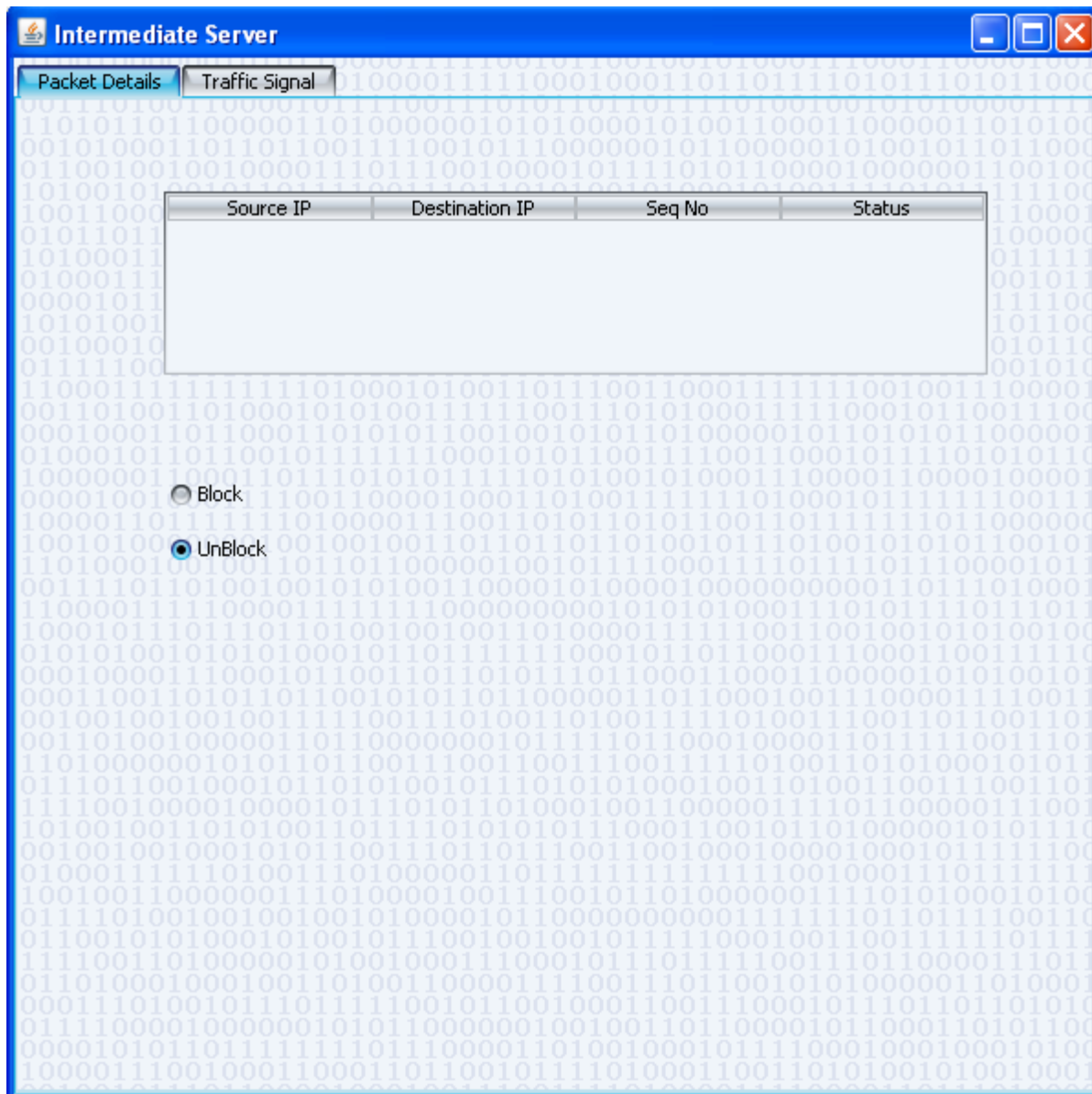
```

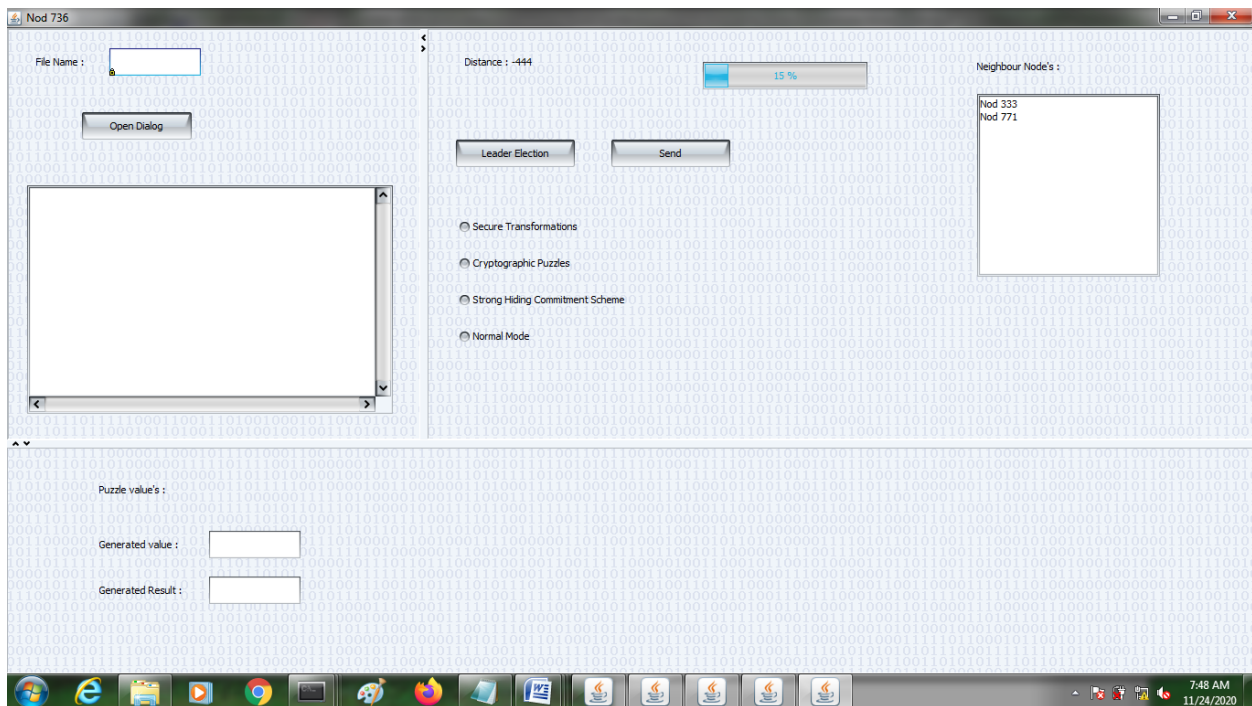
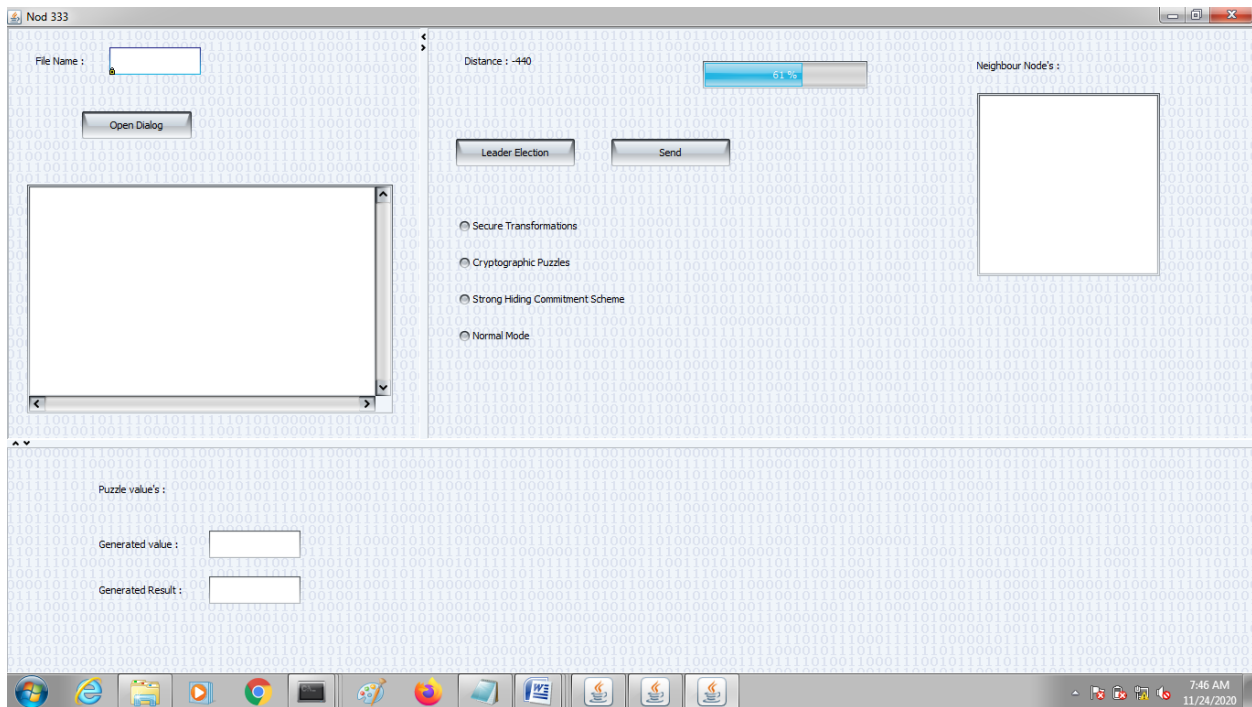
## SCREEN SHOTS

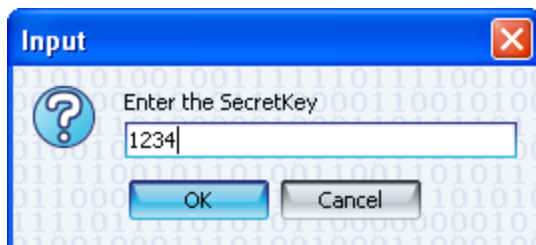
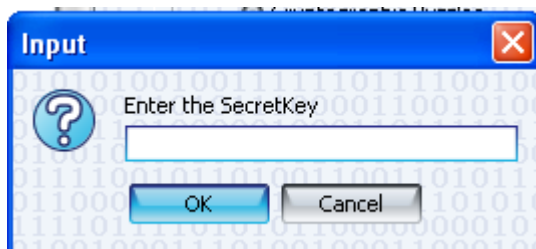
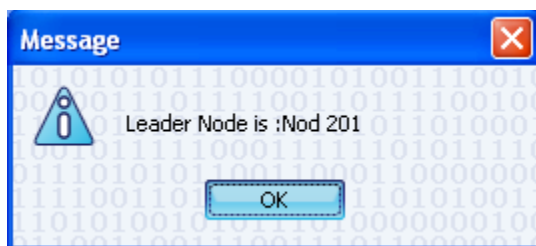
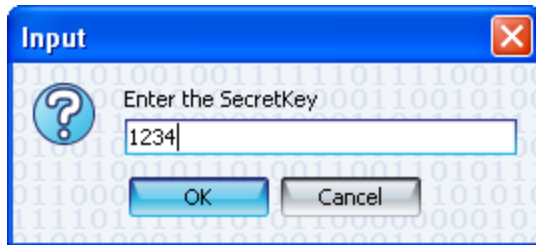
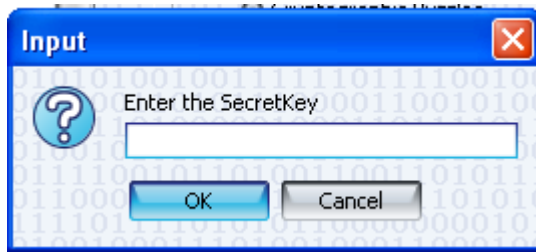
## SCREEN SHOTS












Intermediate Server

Packet Details

Traffic Signal

Source IP	Destination IP	Seq No	Status
192.168.0.1	127.0.0.1	1	UnBlock

☐ Block

☒ Unblock

**Server**

Received Message :

Hi Sample message for jamming attacks project

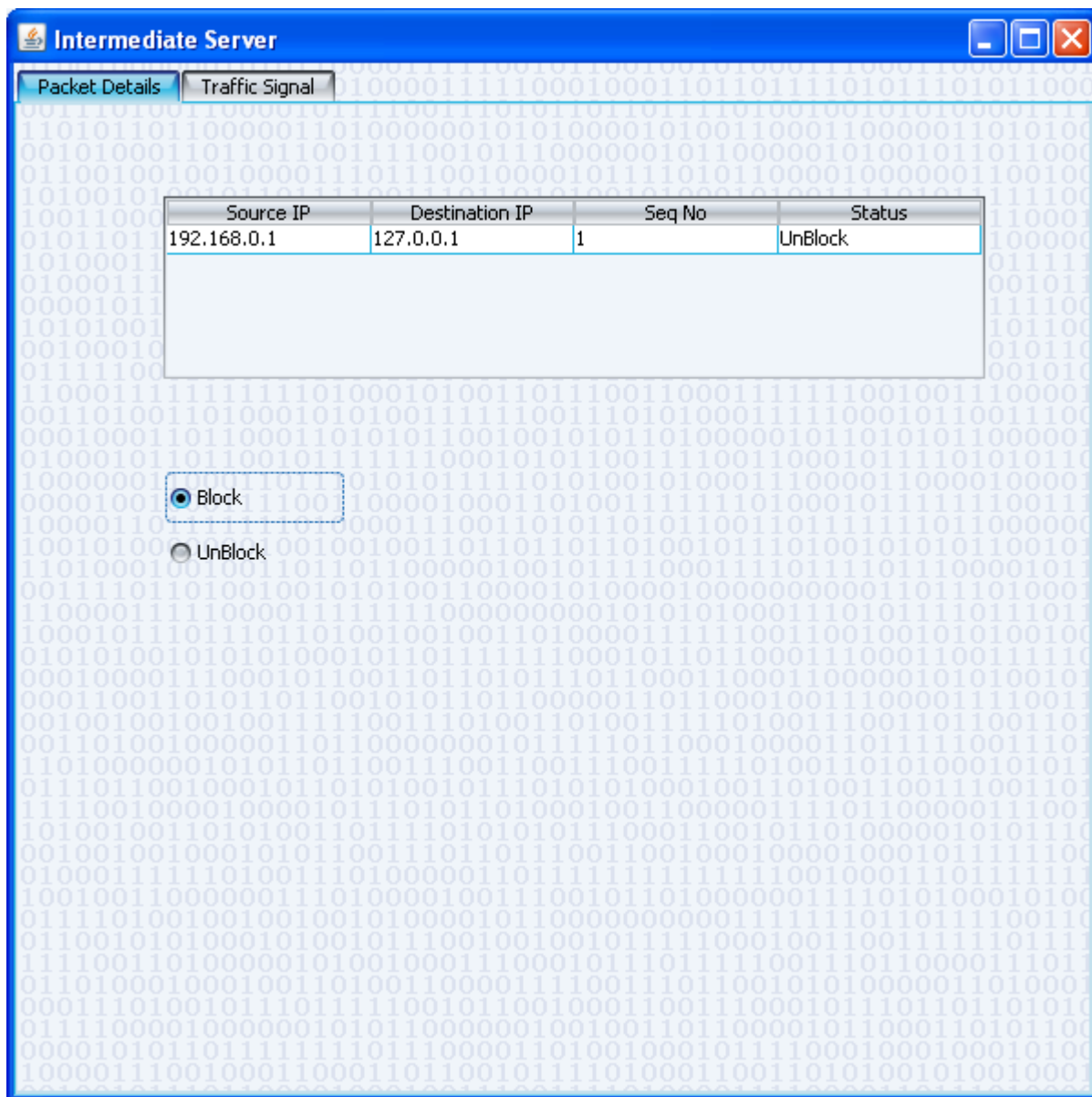
Technical Details :

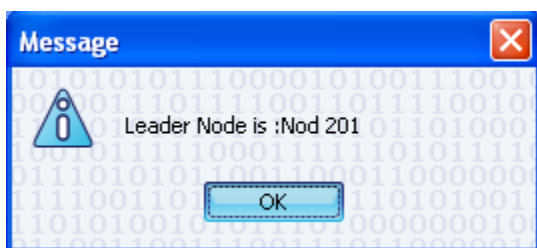
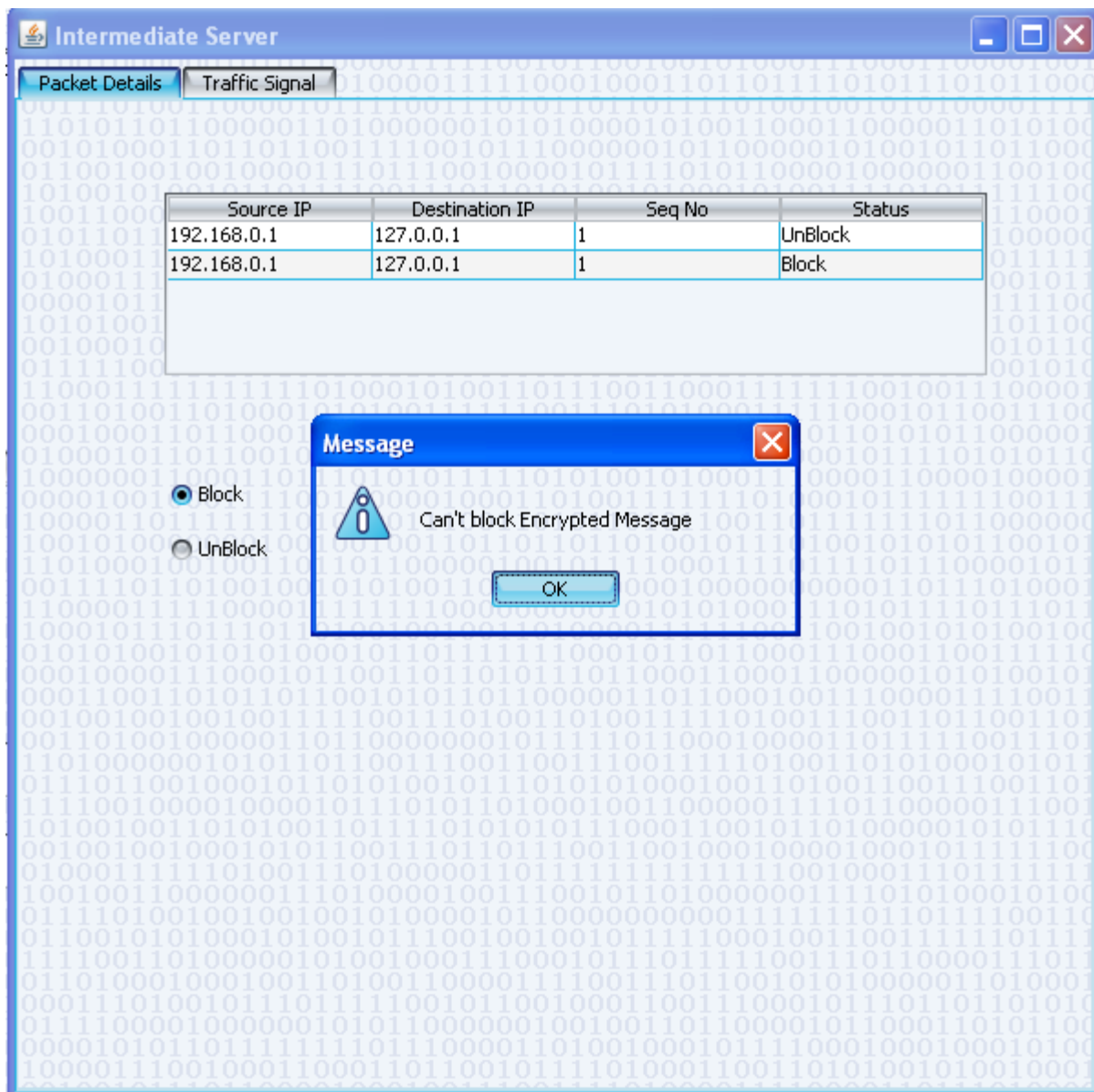
Algorithm Type	Secret key	No of Packets	Secret keypath	Issued Time	Resolved Time
SHCS	1234	1	secret.dat	NA	NA

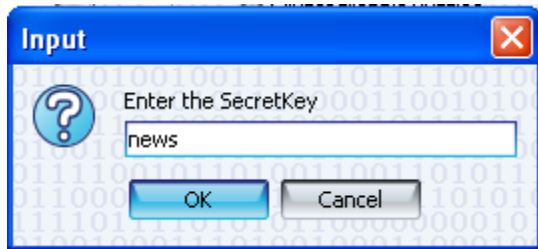
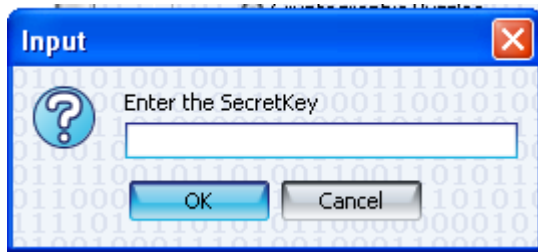
Puzzle value's :

Generated value :

Generated Result :









Server

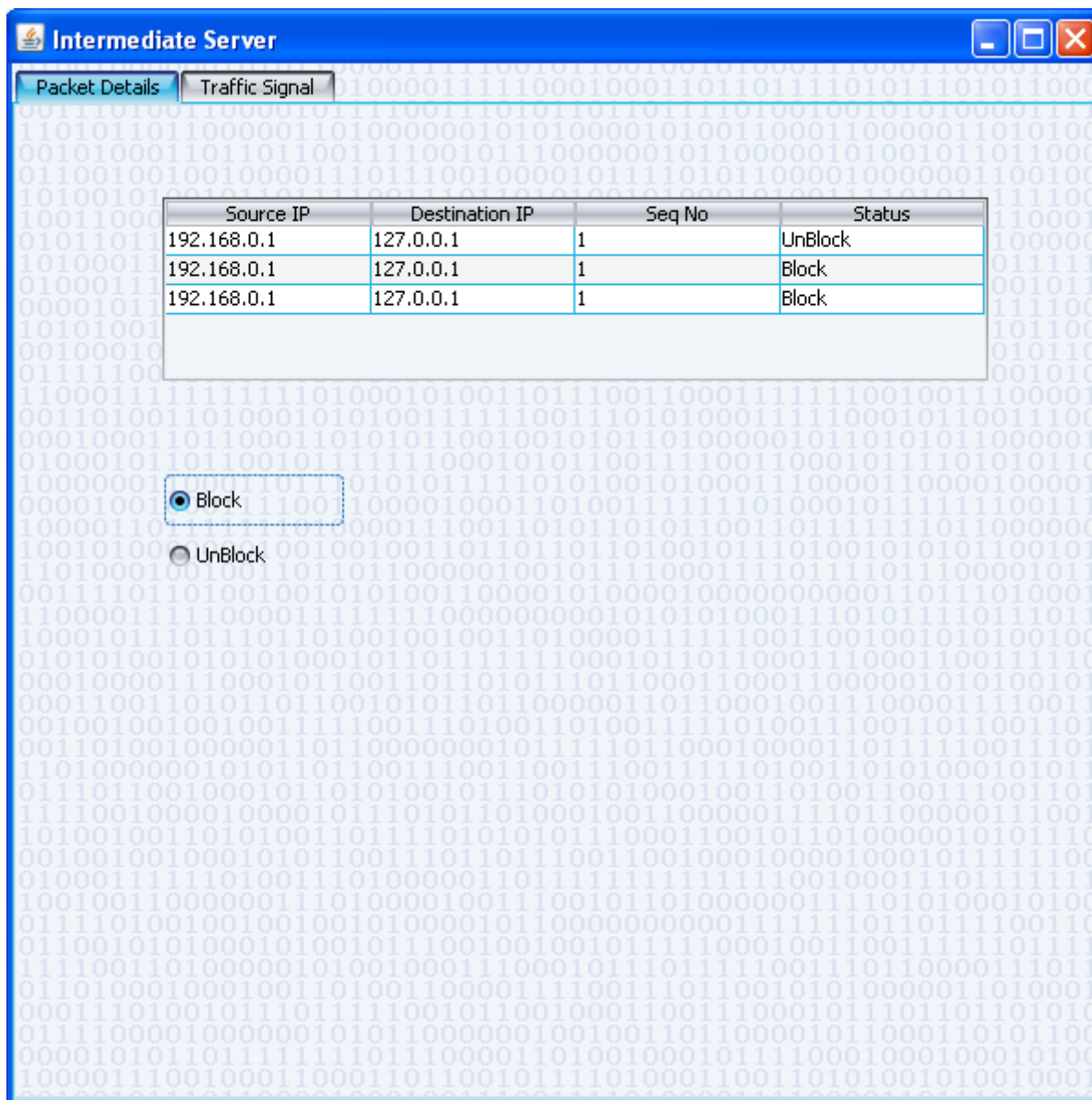
Received Message :

Hi Sample message for jamming attacks project

Puzzle value's :
Generated value :
Generated Result :

Technical Details :

Algorithm Type	Secret key	No of Packets	Secret keypath	Issued Time	Resolved Time
SHCS	1234	1	secret.dat	NA	NA
SHCS	news	1	secret.dat	NA	NA



**Server**

Received Message :

Hi Sample message for jamming attacks project crptogra

Technical Details :

Algorithm Type	Secret key	No of Packets	Secret keypath	Issued Time	Packet's order
SHCS	1234	1	secret.dat	NA	NA
SHCS	news	1	secret.dat	NA	NA
CPHS	$10 + 10 * 5 + 2$	1	secret.dat	60secs	NA

Puzzle value's :  $10 + 10 * 5 + 2$

Generated value :  $10 + 10 * 5 + 2$

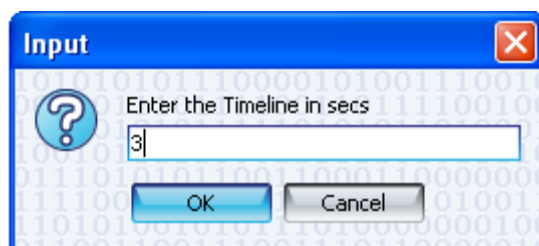
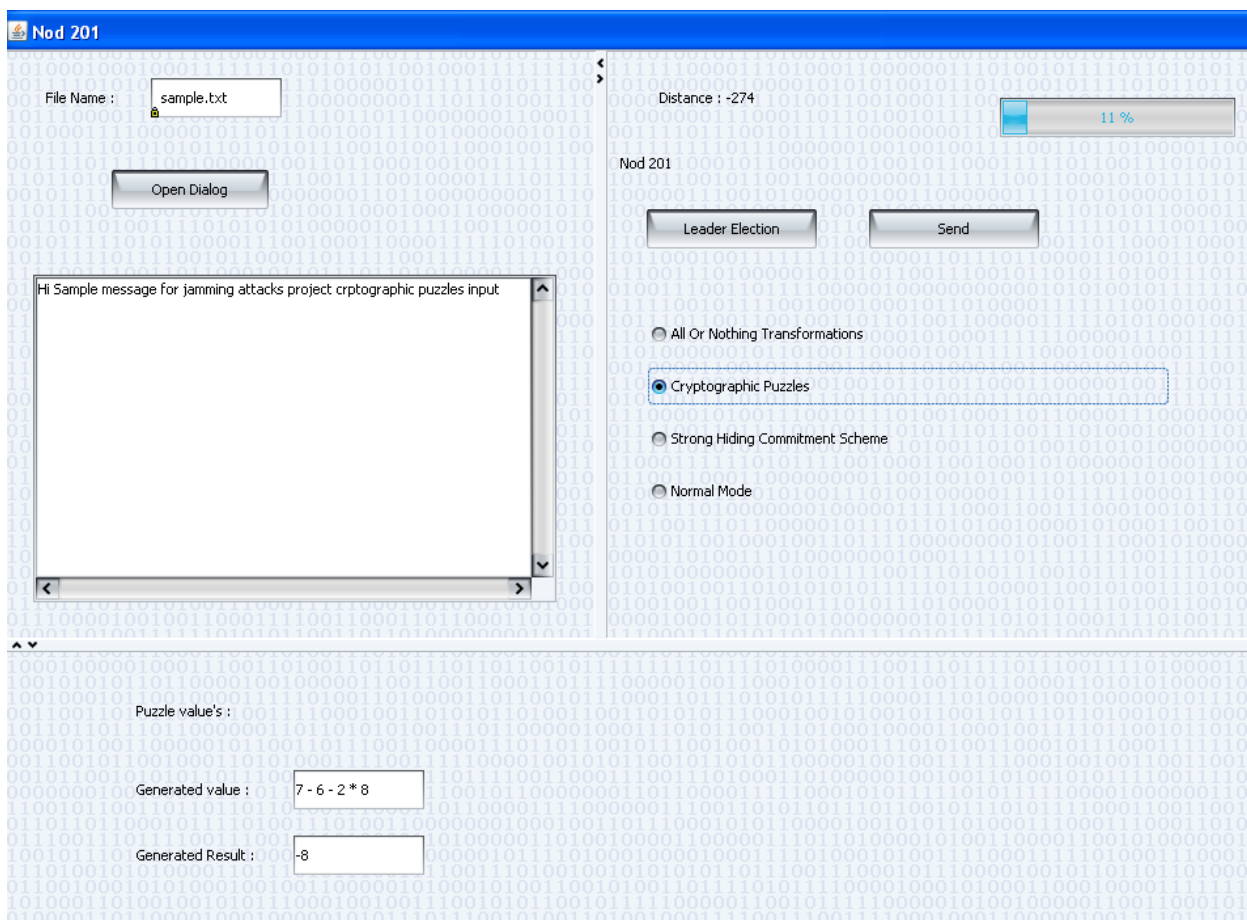
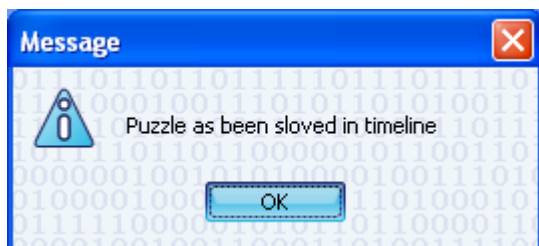
Generated Result : 0

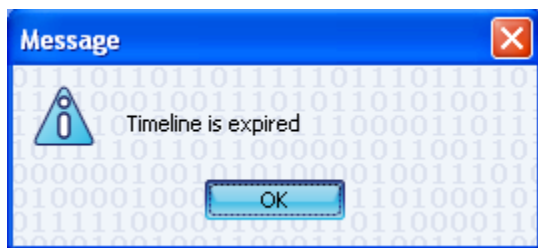
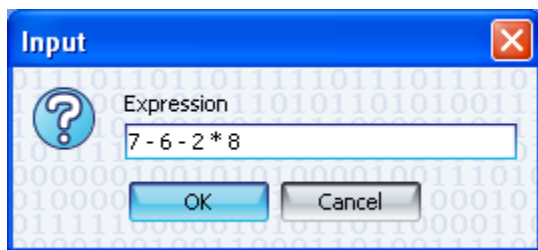
**Input**

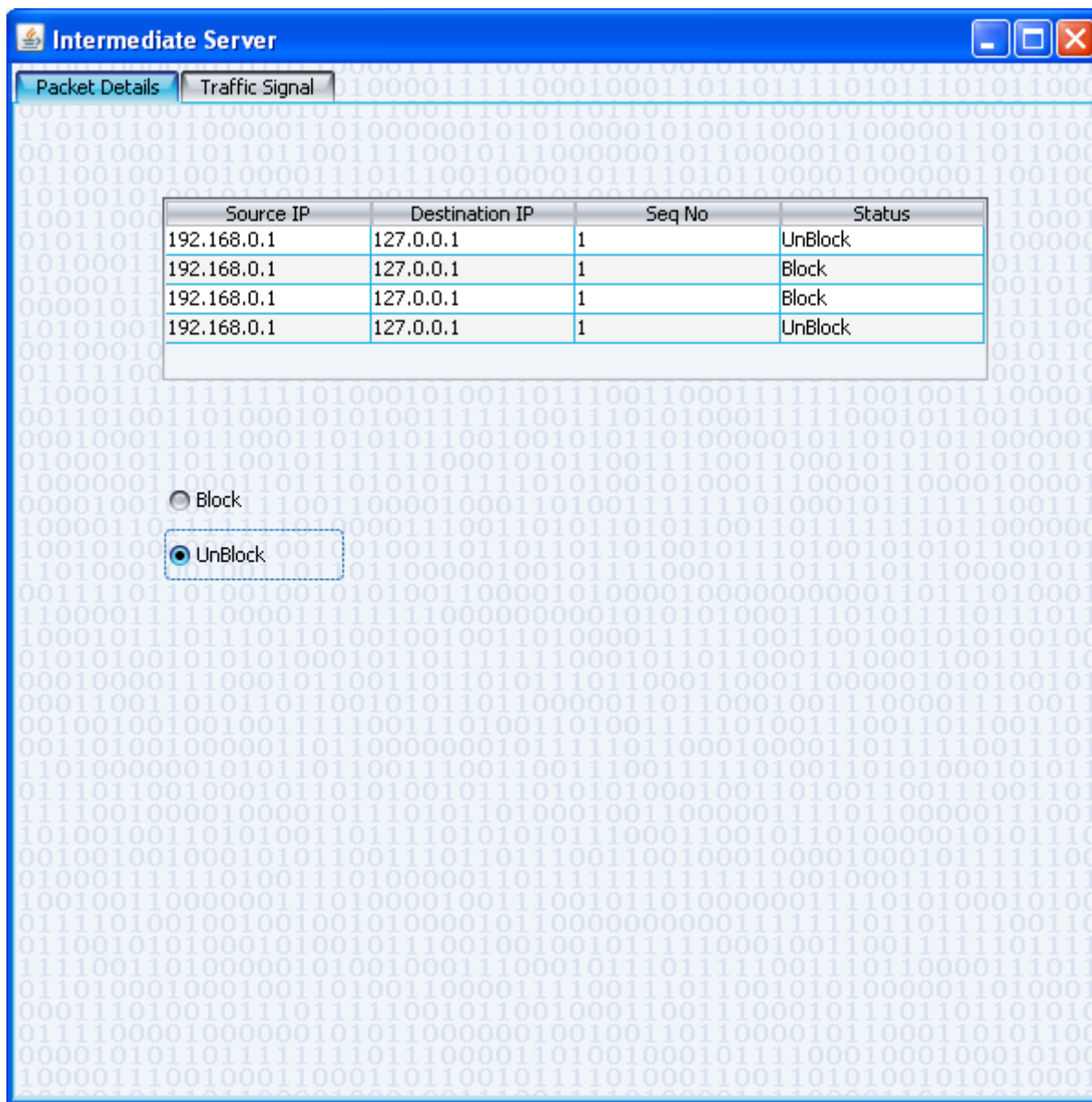
Expression

**Input**

Expression







**Server**

Received Message :

Hi Sample message for jamming attacks project crptograp

Technical Details :

Algorithm Type	Secret key	No of Packets	Secret keypath	Issued Time	Packet's order
SHCS	1234	1	secret.dat	NA	NA
SHCS	news	1	secret.dat	NA	NA
CPHS	$10 + 10 * 5 + 2$	1	secret.dat	60secs	NA

Puzzle value's :  $7 - 6 - 2 * 8$ 
  
Generated value : 
  
Generated Result :

Input

Enter the Timeline in secs

OK


Cancel

Message


Leader Node is :Nod 201

OK


**Input** ✕

 Expression


**Input** ✕

 Expression


**Message** ✕

 Incorrect solution

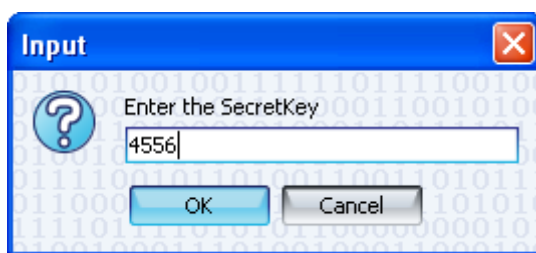
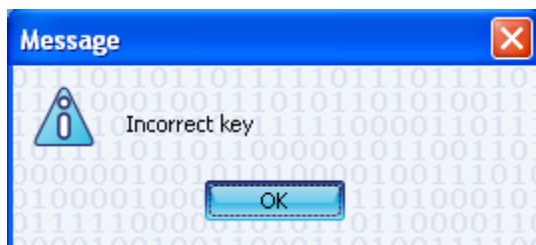
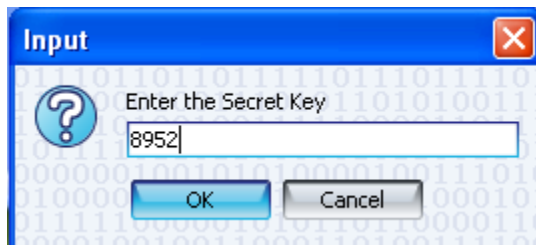
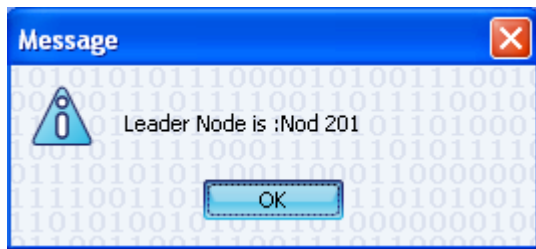
**Input** ✕

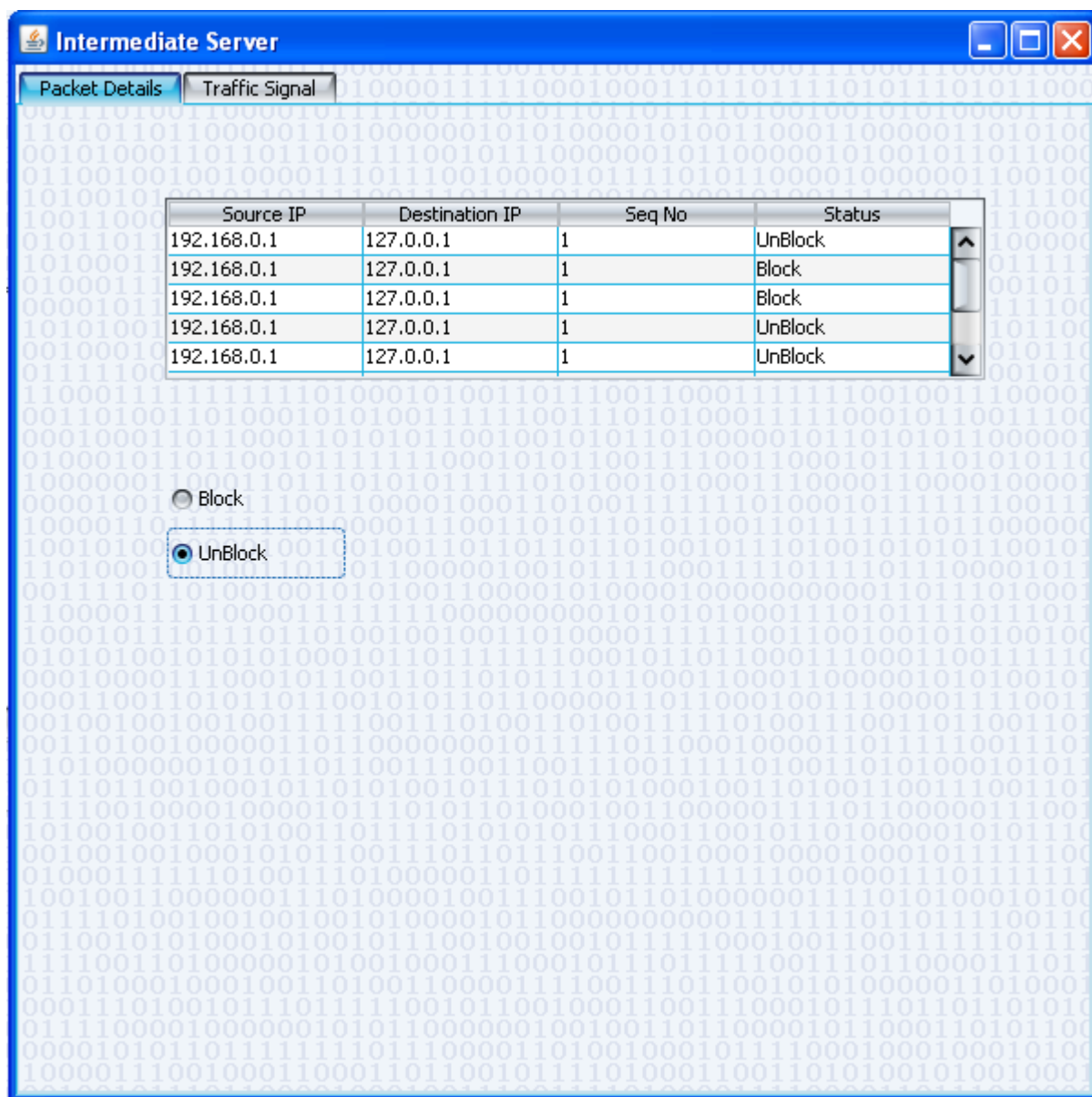
 Enter the SecretKey

**Input** ✕

 Enter the SecretKey







**Server**

Received Message :

Hi Sample message for jamming attacks project all or not

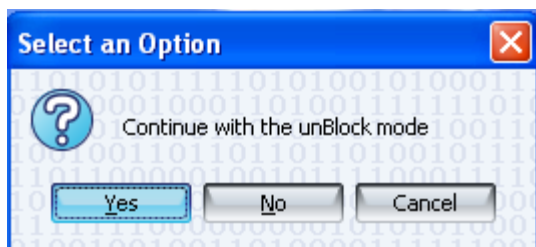
Technical Details :

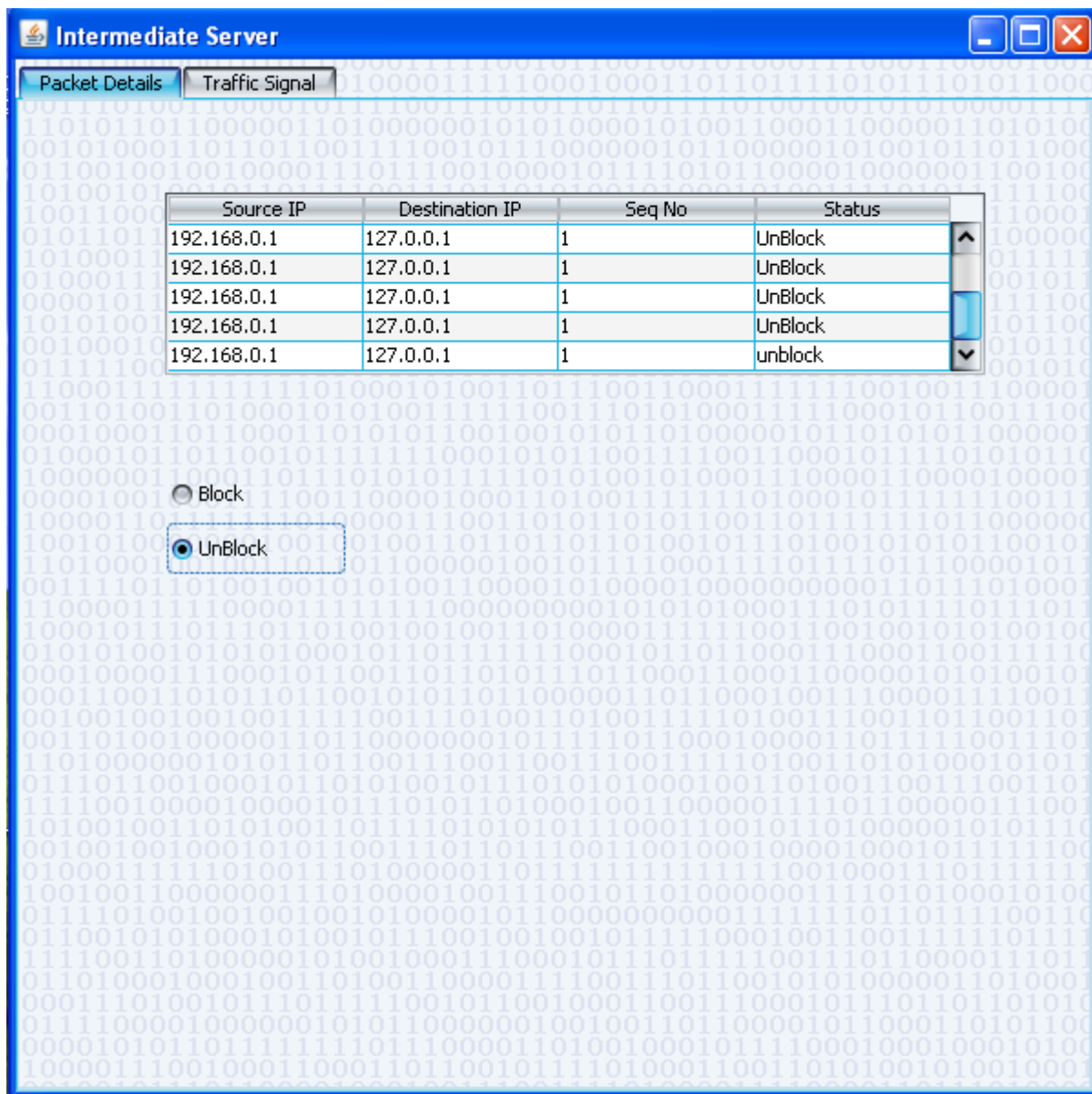
Algorithm Type	Secret key	No of Packets	Secret keypath	Issued Time	Packet's order
SHCS	1234	1	secret.dat	NA	NA
SHCS	news	1	secret.dat	NA	NA
CPHS	$10 + 10 * 5 + 2$	1	secret.dat	60secs	NA
AONT	4556	1	secret.dat	NA	0

Puzzle value's :  $7 - 6 - 2 * 8$

Generated value :

Generated Result :





Server

Received Message :

Hi Sample message for jamming attacks project normal m

Technical Details :

Blocked IP	No of Packets	Blocked Packets
127.0.0.1	1	0

Puzzle value's :

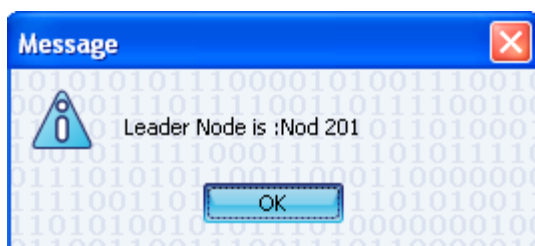
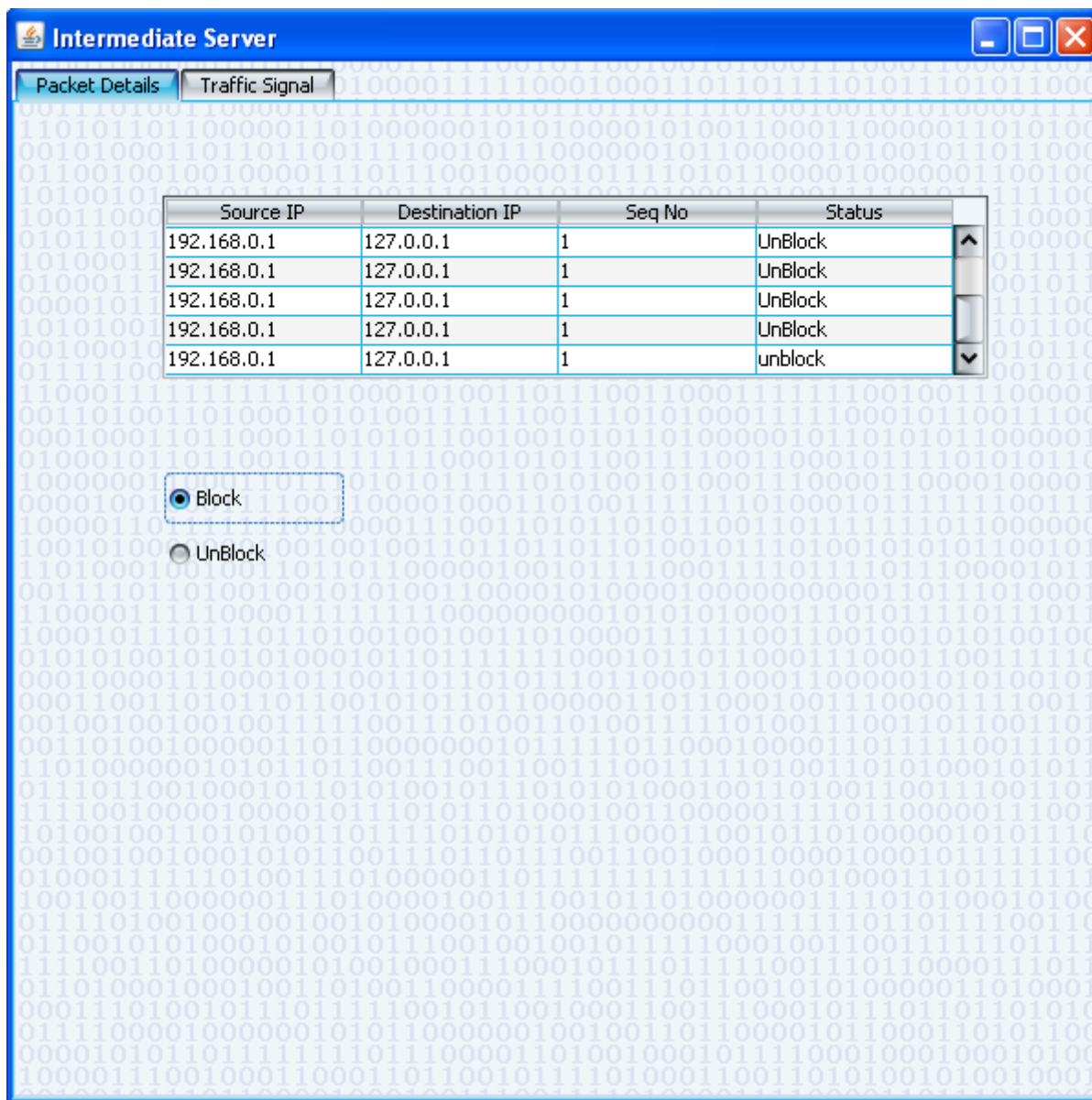
7 - 6 - 2 \* 8

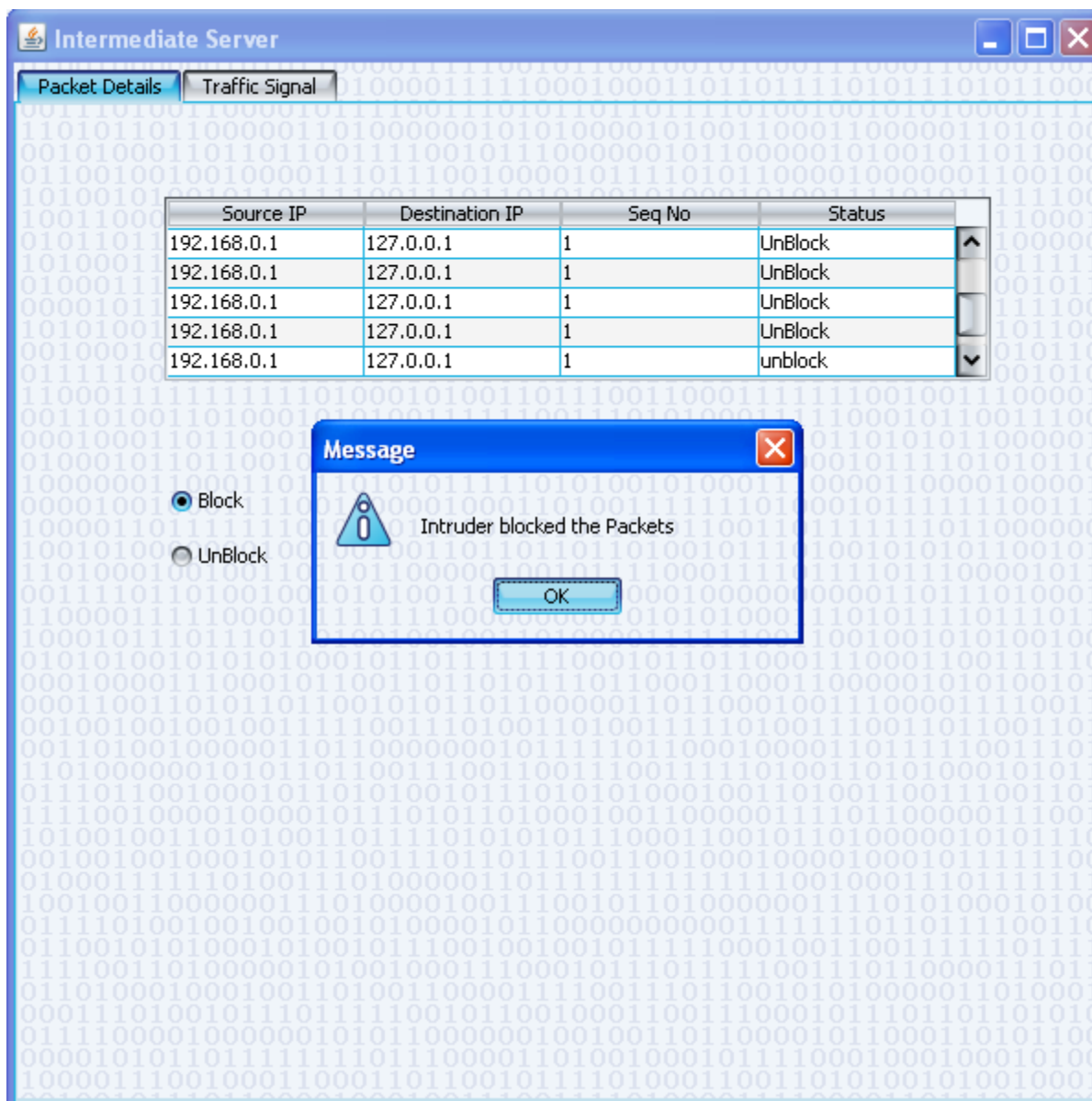
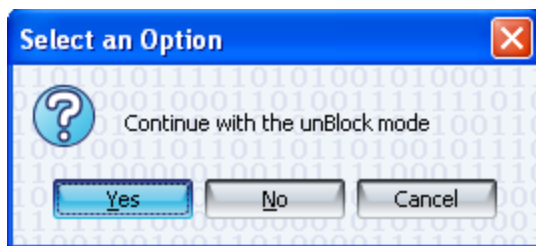
Generated value :

7 - 6 - 2 \* 8

Generated Result :

0





Server

Received Message :

Hi Sample message for jamming attacks project normal m

Puzzle value's :  
7 - 6 - 2 \* 8

Generated value :  
7 - 6 - 2 \* 8

Generated Result :  
0

Technical Details :

Blocked IP	No of Packets	Blocked Packets
127.0.0.1	1	0
127.0.0.1	1	0