1. **INTRODUCTION**

Recent increases in mobile data usage and the emergence of new applications drive the motivation to move the 3GPP into the fourth generation of cellular wireless technology. In response, designers of the 3GPP Long Term Evolution/System Architecture Evolution (LTE/SAE) system have announced the Evolved Packet System (EPS) as the fourth generation of the 3GPP mobile network. The access network used in the EPS network improves radio access technologies of the 3GPP mobile networks so as to offer a higher data rate with low latency. The EPS is also designed to support flat Internet Protocol (IP) connectivity and full interworking with heterogeneous radio access networks and service providers. This architectural design decision brings to the fore implications of LTE/SAE for security. The flat all-IP architecture allows all radio access protocols to terminate in one node called evolved NodeB (eNodeB). In the Universal Mobile Telecommunications System (UMTS), the functionality of eNodeB was divided into NodeB and the Radio Network Controller (RNC). The placement of the radio access protocols in eNodeB makes them vulnerable to unauthorized access because eNodeB is located in unattended place.

Further, internetworking with heterogeneous radio access networks exposes the vulnerability of these networks to direct external threats and carries grave implications for LTE security. The unique characteristics of LTE/SAE gave rise to a number of features in the design of the security mechanism in the EPS network. Of these, key management in handovers and minimizing the security risk involved is the focus of this paper. The main threat to handover key management is that an attack will compromise session keys in a base station. Handover key management typically alleviates this threat through separation of the session keys in a handover between base stations. This separation keeps a session key compromised in one base station from compromising another base station; in other words, the goal is to keep security breaches as local as possible. For reasons of efficiency, handover preparations in LTE/ SAE do not involve the core network. Source eNodeB provides a session key to target eNodeB for use after the handover. In this way, the core network does not need to maintain a state of individual User Equipment (UE).

In this design, handing over an unchanged session key would permit target eNodeB to know which session key the source eNodeB used. To prevent this, the source eNodeB computes a new session key by applying a one-way function to a current session key. This ensures backward key separation in the handover. However, backward key separation blocks an eNodeB only from deriving past session keys from the current session key. Otherwise, this eNodeB would know all session keys used in further sessions in a whole chain of handovers. As a consequence, forward key separation was introduced to ensure that network elements add fresh materials to the process of creating a new session key for the next serving eNodeB. The current eNodeB, unaware of this additive, would be unable to derive the next key. We were able to demonstrate that, under certain circumstances, handover key management fails to ensure forward key separation against a variant attack by a rogue base station; such an attack is herein referred to as a desynchronization attack. A desynchronization attack prevents a target eNodeB from maintaining the freshness of the handover key. The vulnerability of this synchronization to disruption represents a potential security flaw in handover key management that could allow an adversary to compromise all future keys between a specific user and subsequent eNodeBs. This attack may continue until the next update of the root key when handover key materials are generated from scratch instead of by derivation from the previous key. At this point, a potentially devastating effect through a compromised key comes to an end. Without delving into the technical challenges of a specific solution to prevent a desynchronization attack, the most practical remedy is to periodically refresh the root key. A very short-term root key seems an intuitive solution to minimizing the impact of a compromised key.

However, frequent refreshing is not considered the best operational choice because of the signaling load that such root key updating imposes. On the other hand, the longer the update interval the more packets are exposed to a desynchronization attack. The key question network operators and service providers might have is how to effectively choose a root key update interval that is the best balance between the signaling load and the number of user data packets exposed to attack because of a compromised handover key. Unfortunately, because this value is so dependent on time and place, a universally acceptable interval does not exist. Nor are there any proven ways to arrive at acceptable tradeoffs appropriate to different circumstances. In the face of this threat to the next generation of cellular networks, the motivation of this paper is to determine how to formulate this value to fit the circumstances of time and place. As a first step toward a formula for an acceptable tradeoff, we diagramed the timing of handover key management in terms of the root key update interval as a way to measure the period during which a compromised key is operative. We then investigated a mathematical model to measure the expected operative period of the compromised key and to represent the expected value of the signaling load and volume of compromised packets during this period.

Our methodology permits optimal management of the root key update interval according to network policies. This optimal interval is a value that minimizes the signaling traffic overhead required to update the root key while simultaneously limiting the volume of packets exposed to the compromised key.

The main contributions of this paper are threefold:

1) We identified flaws in the handover key management of the EPS security mechanism.

2) We designed a promising mathematical model for the EPS handover key management to measure the effect of a compromised key.

3) We investigated the performance criteria (e.g., user mobility, network topology, and so on) involved in selecting an optimal operational point for key updating. Extensive simulation results validate the analytical model and reveal how the optimal key update interval changes in practice.

1. **LITERATURE REVIEW**

**2.1 PAPERS REFERED:**

Honey Encryption (Encryption beyond the Brute-Force Barrier)

AUTHOR: [Ari Juels](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Ari%20Juels.QT.&newsearch=true) ,[Thomas Ristenpart](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Thomas%20Ristenpart.QT.&newsearch=true)

PUBLISHED: [IEEE Security & Privacy](http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8013) (Volume:12 ,  [Issue: 4](http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=6876237)), July-Aug. 2014

Honey encryption(HE), a simple, general approach to encrypting messages using low min-entropy keys such as passwords. HE is designed to produce a ciphertext which, when decrypted with any of a number of *incorrect* keys, yields plausible-looking but bogus plaintexts called *honey messages*. A key benefit of HE is that it provides security in cases where too little entropy is available to withstand brute-force attacks that try every key; in this sense, HE provides security beyond conventional brute-force bounds.

Using Cloud Computing to Implement a Security Overlay Network:-

AUTHOR: [Khaled Salah](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Khaled%20Salah.QT.&newsearch=true) ,[Jose M. Alcaraz Calero](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Jose%20M.%20Alcaraz%20Calero.QT.&newsearch=true) , [Sherali Zeadally](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Sherali%20Zeadally.QT.&newsearch=true) , [Sameera Al-Mulla](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Sameera%20Al-Mulla.QT.&newsearch=true) ; [Mohammed Alzaabi](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Mohammed%20Alzaabi.QT.&newsearch=true)

PUBLISHED: [IEEE Security & Privacy](http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8013) (Volume:11 ,  [Issue: 1](http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=6427803)), Jan.-Feb. 2013

This article proposes and analyzes a general cloud-based security overlay network that can be used as a transparent overlay network to provide services such as intrusion detection systems, antivirus and anti spam software, and distributed denial-of-service prevention. The authors analyze each of these in-cloud security services in terms of resiliency, effectiveness, performance, flexibility, control, and cost.

Biometric Authentication on Mobile Devices:-

AUTHOR: [Marlies Rybnicek](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Marlies%20Rybnicek.QT.&newsearch=true) ,[Christoph Lang-Muhr](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Christoph%20Lang-Muhr.QT.&newsearch=true) , [Daniel Haslinger](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Daniel%20Haslinger.QT.&newsearch=true)

PUBLISHED: 4-8 Aug. 2014

We examine three biometric authentication modalities – voice, face and gesture – as well as password entry, on a mobile device, to explore the relative demands on user time, effort, error and task disruption. Our laboratory study provided observations of user actions, strategies, and reactions to the authentication methods. Face and voice biometrics conditions were faster than password entry. Speaking a PIN was the fastest for biometric sample entry, but short-term memory recall was better in the face verification condition.

Improving the Security of Cryptographic Protocol Standards:-

AUTHOR: [David Basin](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.David%20Basin.QT.&newsearch=true), [Cas Cremers](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Cas%20Cremers.QT.&newsearch=true) , [Kunihiko Miyazaki](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Kunihiko%20Miyazaki.QT.&newsearch=true) , [Sasa Radomirovic](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Sasa%20Radomirovic.QT.&newsearch=true) , [Dai Watanabe](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Dai%20Watanabe.QT.&newsearch=true)

PUBLISHED: [IEEE Security & Privacy](http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8013) (Volume:13 ,  [Issue: 3](http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=7118073)),May-June 2015

Despite being carefully designed, cryptographic protocol standards often turn out to be flawed. Integrating unambiguous security properties, clear threat models, and formal methods into the standardization process can improve protocol security

LTE/SAE Security Issues on 4G Wireless Networks:-

AUTHOR: [Anastasios N. Bikos](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Anastasios%20N.%20Bikos.QT.&newsearch=true) ,[Nicolas Sklavos](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Nicolas%20Sklavos.QT.&newsearch=true)

PUBLISHED: [IEEE Security & Privacy](http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8013) (Volume:11 ,  [Issue: 2](http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=6493314)), March-April 2013

An overview of the potential security issues that can occur in the deployment of the Long-Term Evolution/System Architecture Evolution protocol in emerging 4G wireless technologies gives a snapshot of the current state of the art.

Along with 3G LTE - Long Term Evolution that applies more to the radio access technology of the cellular telecommunications system, there is also an evolution of the core network known as SAE –System Architecture Evolution. This new architecture has been developed to provide a considerably higher level of performance that is in line with the requirements of LTE.

As a result it is anticipated that operators will commence introducing hardware conforming to the new System Architecture Evolution standards so that the anticipated data levels can be handled when 3G LTE is introduced.

The new SAE, System Architecture Evolution has also been developed so that it is fully compatible with LTE Advanced, the new 4G technology. Therefore when LTE Advanced is introduced, the network will be able to handle the further data increases with little change.

The SAE System Architecture Evolution offers many advantages over previous topologies and systems used for cellular core networks. As a result it is anticipated that it will be wide adopted by the cellular operators.

Because of such vast advantages over previous topologies and also the fact that the LTE/SAE system will be used by the majority of stakeholders, we have chosen this topic over the above mentioned topics. We believe that any fruitful research on the chosen topic will be relevant and useful for a considerable amount of time. Being a “burning topic” or a “trending topic”, the research will be more interesting and also relatable as we use these services on a daily basis

**2.2 TERMS**

Encryption: A process of converting plain text into cipher text is called as Encryption. This process requires two things- an encryption algorithm and a key. Algorithm means the technique that has been used in encryption. Encryption of data takes place at the sender side.

Authentication: **Authentication** is a process in which the credentials provided are compared to those on file in a database of authorized users' information on a local operating system or within an **authentication** server. If the credentials match, the process is completed and the user is granted authorization for access.

The Kasumi Algorithm: The first ciphering algorithm for the LTE standard, the Kasumi algorithm, is mainly a block cipher algorithm that uses a key size of 128 bits. The algorithm utilizes two mapping functions to produce the cipher- text, which are called *S-boxes*. Kasumi was specifically designed as a building block for the UMTS encryption algorithms (UEA1) and integrity algorithms (UIA1).

SNOW 3G:SNOW 3G was designed as a second cryptographic solution in response to the appearance of newer forms of attacks—algebraic attacks—that would decrease Kasumi-based algorithms’ security. Similar to the case of Kasumi, some changes were made to the original SNOW .0 to adapt it to the requirements of the demanding 3G environment and defend itself successfully against the newly discovered algebraic attacks. SNOW 3G is used as the core component of both UEA2 and UIA2.

The Milenage Algorithm: The Milenage encryption algorithm is the third 3G security algorithm deployed in LTE that uses a core function of a block cipher in which both block size and key size are 128 bits. Here, we can use the basic form of the Advanced Encryption Standard encryption algorithm as the core function.6

The ZUC Algorithm: In addition to the previous 3G cryptographic algorithms, the ETSI SAGE task force, together with Chinese cryptography experts, have already started the design work for a third algorithm pair specifically for 4G security. This beyond-3G cryptographic component is the ZUC algorithm, and it will be the core cipher block for UEA3/UIA3 set.

3GPP: **The 3rd Generation Partnership Project** (**3GPP**) is a collaboration between groups of telecommunications associations, known as the Organizational Partners. The initial scope of 3GPP was to make a globally applicable third-generation (3G) mobile phone system specification based on evolved Global System for Mobile Communications (GSM) specifications within the scope of the International Mobile Telecommunications-2000 project of the International Telecommunication Union (ITU). The scope was later enlarged to include the development and maintenance of:

* GSM and related "2G" and "2.5G" standards including GPRS and EDGE
* UMTS and related "3G" standards including HSPA
* LTE and related "4G" standards
* An evolved IP Multimedia Subsystem (IMS) developed in an access independent manner

EAP-AKA:Extensible Authentication Protocol Method for Universal Mobile Telecommunications System (UMTS) Authentication and Key Agreement (EAP-AKA), is an EAP mechanism for authentication and session key distribution using the UMTS Subscriber Identity Module (USIM). EAP-AKA is defined in RFC 4187.

Mobility Management Entity (MME): The MME is the main control node for the LTE SAE access network, handling a number of features:

* + Idle mode UE tracking
  + Bearer activation / de-activation
  + Choice of SGW for a UE
  + Intra-LTE handover involving core network node location
  + Interacting with HSS to authenticate user on attachment and implements roaming restrictions

Serving Gateway (SGW): The Serving Gateway, SGW, is a data plane element within the LTE SAE. Its main purpose is to manage the user plane mobility and it also acts as the main border between the Radio Access Network, RAN and the core network. The SGW also maintains the data paths between the eNodeBs and the PDN Gateways. In this way the SGW forms a interface for the data packet network at the E-UTRAN.Also when UEs move across areas served by different eNodeBs, the SGW serves as a mobility anchor ensuring that the data path is maintained.

PDN Gateway (PGW): The LTE SAE PDN gateway provides connectivity for the UE to external packet data networks, fulfilling the function of entry and exit point for UE data. The UE may have connectivity with more than one PGW for accessing multiple PDNs.

Policy and Charging Rules Function (PCRF):This is the generic name for the entity within the LTE SAE EPC which detects the service flow, enforces charging policy. For applications that require dynamic policy or charging control, a network element entitled the Applications Function, AF is used.

1. **SYSTEM ANALYSIS**

**3.1 EXSISTING SYSTEM:**

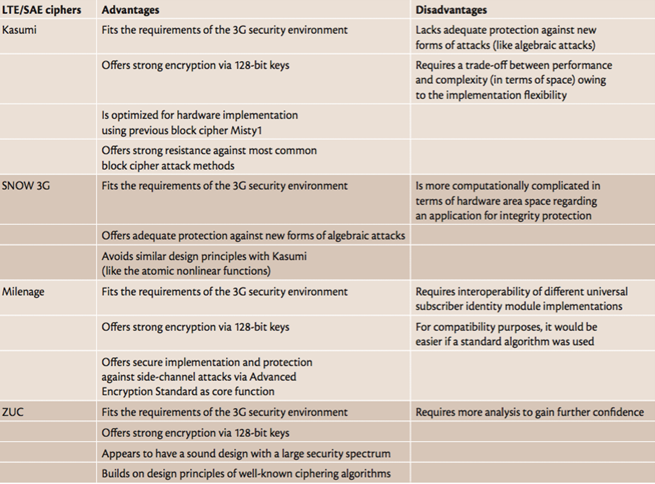
Existing analyzes the authentication and key agreement protocol adopted by Universal Mobile Telecommunication System (UMTS), an emerging standard for third-generation (3G) wireless communications. The protocol, known as 3GPP AKA, is based on the security framework in GSM and provides significant enhancement to address and correct real and perceived weaknesses in GSM and other wireless communication systems.

3GPP AKA protocol is vulnerable to a variant of the so-called false base station attack. The vulnerability allows an adversary to redirect user traffic from one network to another. It also allows an adversary to use authentication vectors corrupted from one network to impersonate all other networks. Moreover, we demonstrate that the use of synchronization between a mobile station and its home network incurs considerable difficulty for the normal operation of 3GPP AKA.

Security problems in the 3GPP AKA, we then present a new authentication and key agreement protocol which defeats redirection attack and drastically lowers the impact of network corruption. The protocol, called AP-AKA, also eliminates the need of synchronization between a mobile station and its home network. AP-AKA specifies a sequence of multiple flows.

**3.1.1 LTE/SAE 3G Cryptographic Algorithms:-**

Although 3G security specifications have already been standardized for the LTE protocol, extensible research is still carried out by the European Telecommunications Standards Institute (ETSI)—experts for the 4G standard of LTE/SAE. The Security Algorithms Group of Experts (SAGE) task force analyzed the basic 3G algorithms for encryption and integrity protection for the LTE network standard. Table 1 presents the advantages and disadvantages of each.



**Table 3.1.2 Advantages and disadvantages of the LTE/SAE encryption schemes**

**3.2 PROBLEMS WITH EXSISTING SYSTEM**

Several EPS-specific threats that concern the whole EPS architecture and trust model have emerged with the characteristics of radio interface—thus posing great risks for the standards integrity environment.

We further classify the main threat and risk categories that degrade the EPS security reliability as follows:

* Threats against user identity and privacy.
* Threats of UE/USIM tracking.
* Threats related to base stations and handovers.
* Threats related to broadcast or multicast signaling.
* Threats related to denial of service (DoS).
* Threats against manipulation of control plane data.
* Threats of unauthorized access to the network.
* Compromise of eNB credentials as well as physical attacks on an eNB.
* Protocol attacks on an eNB.
* Attacks on the core network, including eNB location-based attacks.

**3.3 PROPOSED SYSTEM**

Our proposed method an unchanged session key would permit target eNodeB to know which session key the source eNodeB used. To prevent this, the source eNodeB computes a new session key by applying a one-way function to a current session key. This ensures backward key separation in the handover. However, backward key separation blocks an eNodeB only from deriving past session keys from the current session key. Otherwise, this eNodeB would know all session keys used in further sessions in a whole chain of handovers. As a consequence, forward key separation was introduced to ensure that network elements add fresh materials to the process of creating a new session key for the next serving eNodeB. The current eNodeB, unaware of this additive, would be unable to derive the next key.

The main contributions of this paper are threefold:

1) We identified flaws in the handover key management of the EPS security mechanism;

2) We designed a promising mathematical model for the EPS handover key management to measure the effect of a compromised key;

3) We investigated the performance criteria (e.g., user mobility, network topology, and so on) involved in selecting an optimal operational point for key updating.

**3.4 SECURITY ENHANCEMENTS IN LTE/SAE STANDARD**

For most security attacks, many crucial security features came along with the security standardization of the LTE/SAE. Many of those requirements led to the EPS security architecture being quite different from the 3G security architecture. Some of the design decisions are as follows:

* Permanent security association
* New key hierarchy in EPS
* Need for mutual authentication mechanisms
* Trusted environment and secure execution
* DoS protection of network
* User privacy
* Authorization

1. **SYSTEM REQUIREMENTS**

**4.1 HARDWARE REQUIREMENTS:-**

# Processor - Pentium –IV or above

* + Speed - 1.1 GHz
  + RAM - 256 MB (min)
  + Hard Disk - 20 GB
  + Key Board - Standard Windows Keyboard
  + Mouse - Two or Three Button Mouse
  + Monitor - SVGA

# 4.2 SOFTWARE REQUIREMENTS:-

* Operating System - Windows XP and above
* Front End - Java JDK 1.6 and above
* Document - MS-Office 2013
* Tools - Eclipse IDE

1. **OVERVIEW OF TECHNOLOGIES AND TOOLS**

**5.1 JAVA**

**5.1.1 Java Programming Language**

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

* Simple
* Architecture neutral
* Object oriented
* Portable
* Distributed
* High performance
* Interpreted
* Multithreaded
* Robust
* Dynamic
* Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed

**5.1.2 The Java Platform**

A platform is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

* The Java Virtual Machine (Java VM)
* The Java Application Programming Interface (Java API)

You’ve already been introduced to the Java VM. It’s the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

**5.1.3 Networking TCP/IP stack:**

## xxx_files/tcp_stack.gif

Fig 5.1.3: The TCP/IP network stack

TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

### IP data-grams: The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

### UDP: UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

### TCP:TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

### **5.1.4 Sockets**:

### A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

#include <sys/types.h>

#include <sys/socket.h>

int socket(int family, int type, int protocol);

Here "family" will be AF\_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

**5.1.5 Addresses:**

* **Internet addresses**: In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

### **Network address**: Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

### **Subnet address:** Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

### **Host address:** 8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

### **Total address:**



Fig 5.1.7 Total address structure

The 32 bit address is usually written as 4 integers separated by dots.

* **Port addresses:**  A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

**5.2 ECLIPSE LUNA IDE**

In [computer programming](https://en.wikipedia.org/wiki/Computer_programming), **Eclipse** is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE). It contains a base[workspace](https://en.wikipedia.org/wiki/Workspace) and an extensible [plug-in](https://en.wikipedia.org/wiki/Plug-in_(computing)) system for customizing the environment. Eclipse is written mostly in[Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and its primary use is for developing Java applications, but it may also be used to develop applications in other programming language. Variouspulgins including: [Ada](https://en.wikipedia.org/wiki/Ada_(programming_language)), [ABAP](https://en.wikipedia.org/wiki/ABAP), [C](https://en.wikipedia.org/wiki/C_(programming_language)), [C++](https://en.wikipedia.org/wiki/C%2B%2B),[COBOL](https://en.wikipedia.org/wiki/COBOL), [Fortran](https://en.wikipedia.org/wiki/Fortran), [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [Julia](https://en.wikipedia.org/wiki/Julia_(programming_language)), [Lasso](https://en.wikipedia.org/wiki/Lasso_(programming_language)), [Lua](https://en.wikipedia.org/wiki/Lua_(programming_language)), [NATURAL](https://en.wikipedia.org/w/index.php?title=NATURAL&action=edit&redlink=1), [Perl](https://en.wikipedia.org/wiki/Perl), [PHP](https://en.wikipedia.org/wiki/PHP), [Prolog](https://en.wikipedia.org/wiki/Prolog), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [R](https://en.wikipedia.org/wiki/R_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language))(including [RubyonRails](https://en.wikipedia.org/wiki/Ruby_on_Rails) framework), [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)), [Clojure](https://en.wikipedia.org/wiki/Clojure), [Groovy](https://en.wikipedia.org/wiki/Groovy_(programming_language)), [Scheme](https://en.wikipedia.org/wiki/Scheme_(programming_language)), and [Erlang](https://en.wikipedia.org/wiki/Erlang_(programming_language)" \o "Erlang (programming language)). It can also be used to develop packages for the software [Mathematica](https://en.wikipedia.org/wiki/Mathematica" \o "Mathematica). Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for PHP, among others.

The initial [codebase](https://en.wikipedia.org/wiki/Codebase) originated from [IBM VisualAge](https://en.wikipedia.org/wiki/IBM_VisualAge). The Eclipse [software development kit](https://en.wikipedia.org/wiki/Software_development_kit) (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules.

Released under the terms of the [Eclipse Public License](https://en.wikipedia.org/wiki/Eclipse_Public_License), Eclipse [SDK](https://en.wikipedia.org/wiki/Software_development_kit) is [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open-source_software)(although it is incompatible with the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License)). It was one of the first IDEs to run under[GNU Classpath](https://en.wikipedia.org/wiki/GNU_Classpath) and it runs without problems under [IcedTea](https://en.wikipedia.org/wiki/IcedTea" \o "IcedTea).

1. **SYSTEM MODELLING**

**6.1 DATA FLOW / USE CASE / FLOW DIAGRAMS**

* The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system
* The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
* DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
* DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

### **6.2 NOTATION**

### Source or destination of data:

External sources or destinations, which may be people or organizations or other entities

### Data Source:

Here the data referenced by a process is stored and retrieved.

### Process:

People, procedures or devices that produce data. The physical component is not identified.

### Data flow:

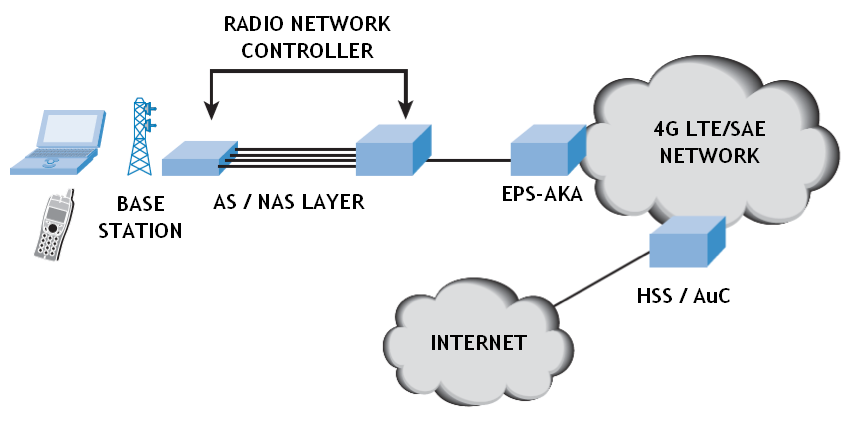
Data moves in a specific direction from an origin to a destination. The data flow is a “packet” of data.

**6.3 MODELING RULES**

There are several common modeling rules when creating DFDs:

1. All processes must have at least one data flow in and one data flow out.
2. All processes should modify the incoming data, producing new forms of outgoing data.
3. Each data store must be involved with at least one data flow.
4. Each external entity must be involved with at least one data flow.
5. A data flow must be attached to at least one process

**6.4 ARCHITECTURE DIAGRAM**

****

**Fig 6.4 The basic architecture of the 4G network**

**Fig 6.4 Proposed architecture diagram**

**6.5 DATAFLOW DIAGRAM**

The entire process of information or data flow takes place in two phases or levels:

* level 0: The first levels focuses mainly on the generation of the secret session key
* level 1:The second levels focuses on the exchange of information between the sender and the receiver via the server using the secret session key generated in level 0

**LEVEL – 0:**

Secret Hash Key

Session Key

Key Management

Random String Gen

Hashing

**Fig 6.5.1: Level-0 of data transfer - secret session key generation**

**LEVEL – 1:**

Secret Key

Secret Key

Session Key

Encrypted Msg by Sess Key

**Level1**

Key Generation

**Fig 6.5.2: Level-1 of data transfer- Data transfer using the secret session key**

**6.6 UML DIAGRAMS**

**Use Case Diagram:**

**Fig 6.6.1 Use Case Diagram for the message transfer**

**Class Diagram:**

Sender

Request ()

Upload ()

String Filename

EPS-AKA

Randomnumber()

Sessionkey()

String Key

Hash Key Generation

Setup ()

KeyDistribution()

String Secretkey

Receiver

Request ()

Download ()

String Filename

**Fig 6.6.2 Class diagram for all the modules in the system**

**Object Interaction Diagram:**

EPS-AKA

Random Key Generation

Session Key Generation

Create Hash Key

Sender

Hash Key matching

Transaction declined

Transaction allowed

Original data

Request

Receiver

Users pre-share secret keys with the TC

**Fig 6.6.3 Object interaction diagram showing the entire process**

1. **IMPLEMENTATION**

**7.1 EPS-AKA:**

EPS network, an Authentication and Key Agreement (EPS-AKA) occurs between a UE and the MME on behalf of the Home Subscriber Server (HSS)/Authentication Center (AuC). The EPS-AKA is the EPS security mechanism to execute:

* Authentication between a UE and an MME on behalf of the HSS/AuC, and
* A key agreement between a UE and an MME as well as between a UE and eNodeB.

Authentication succeeds; the two parties generate the first intermediate key, KASME, from the permanent master key, K. In the course of performing EPS-AKA, the HSS/AuC delivers the first intermediate key to the MME after binding to the serving network identity the evolution to LTE and its flat all-IP core network emphasizes the urgent need for a revision of the trust relationships between operators and network components. Any threats arising from un trusted networks are alleviated in the EPS by a new feature, namely cryptographic network separation.

Network separation tries to isolate the impact of any security breach in the local network and prevent its spillover to other networks. This is achieved by binding any cryptographic keys to the identity of the serving network for which the keys are intended. The UE can ensure that it communicates with the intended serving network by authenticating an identity in the current network. In the UMTS, a UE was unable to authenticate a serving network.

The local master key, KASME, also called the first intermediate key, is valid at a maximum interval determined by the timing of the next EPS-AKA procedure. The UE can choose to invoke the EPS-AKA protocol whenever the serving MME changes because of roaming to another serving network. In the same situation, the UE also can choose to transfer the security context between the old and new MMEs in an effort to lower the overhead of the full EPS-AKA. The UE may, of course, also need to run the EPSAKA protocol periodically without interrupting service.

Hence, the frequency of EPS-AKA runs is rather random or configurable by a network operator.

**7.2 ALGORITHM: Advanced Encryption Standard (AES)**

AES is based on a design principle known as a [substitution-permutation network](https://en.wikipedia.org/wiki/Substitution-permutation_network), combination of both substitution and permutation, and is fast in both software and hardware. Unlike its predecessor DES, AES does not use a [Feistel network](https://en.wikipedia.org/wiki/Feistel_network). AES is a variant of Rijndael which has a fixed [block size](https://en.wikipedia.org/wiki/Block_size_(cryptography)) of 128 [bits](https://en.wikipedia.org/wiki/Bit), and a [key size](https://en.wikipedia.org/wiki/Key_size) of 128, 192, or 256 bits. By contrast, the Rijndael specification *per se* is specified with block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits.

AES operates on a 4×4 [column-major order](https://en.wikipedia.org/wiki/Column-major_order) matrix of bytes, termed the *state*, although some versions of Rijndael have a larger block size and have additional columns in the state. Most AES calculations are done in a special [finite field](https://en.wikipedia.org/wiki/Finite_field_arithmetic).

The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the cipher text. The number of cycles of repetition are as follows:

* 10 cycles of repetition for 128-bit keys.
* 12 cycles of repetition for 192-bit keys.
* 14 cycles of repetition for 256-bit keys.

Each round consists of several processing steps, each containing four similar but different stages, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform cipher text back into the original plaintext using the same encryption key.

**High level description of the algorithm:**

1. Key Expansions—round keys are derived from the cipher key using [Rijndael's key schedule](https://en.wikipedia.org/wiki/Rijndael_key_schedule" \o "Rijndael key schedule). AES requires a separate 128-bit round key block for each round plus one more.

2. Initial Round- Add Round Key—each byte of the state is combined with a block of the round key using bitwise xor.

3.Rounds

* 1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a [lookup table](https://en.wikipedia.org/wiki/Rijndael_S-box).
  2. ShiftRows—a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.
  3. MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
  4. AddRoundKey

4. Final Round (no MixColumns)

* 1. SubBytes
  2. ShiftRows
  3. AddRoundKey.

**7.3 MODULES**

* SERVER CLIENT MODULE
* OPTIMAL KEY MANAGEMENT
* AUTHENTICATION KEY AGREEMENT
* LONG-TERM EVOLUTION SECURITY

**7.3.1 SERVER CLIENT MODULE:**

The client–server model of computing is a [distributed application](https://en.wikipedia.org/wiki/Distributed_application) structure that partitions tasks or workloads between the providers of a resource or service, called [servers](https://en.wikipedia.org/wiki/Server_(computing)), and service requesters, called [clients](https://en.wikipedia.org/wiki/Client_(computing)). Often clients and servers communicate over a [computer network](https://en.wikipedia.org/wiki/Computer_network) on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests. Examples of computer applications that use the client–server model are [Email](https://en.wikipedia.org/wiki/Email), [network printing](https://en.wikipedia.org/wiki/Network_printing), and the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web).

The Client-server characteristic describes the relationship of cooperating programs in an application. The server component provides a function or service to one or many clients, which initiate requests for such services. Servers are classified by the services they provide. For instance, a [web server](https://en.wikipedia.org/wiki/Web_server) serves [web pages](https://en.wikipedia.org/wiki/Web_page) and a [file server](https://en.wikipedia.org/wiki/File_server) serves [computer files](https://en.wikipedia.org/wiki/Computer_file). A shared resource may be any of the server computer's software and electronic components, from [programs](https://en.wikipedia.org/wiki/Computer_program) and [data](https://en.wikipedia.org/wiki/Data_(computing)) to [processors](https://en.wikipedia.org/wiki/Microprocessor) and [storage devices](https://en.wikipedia.org/wiki/Data_storage_device). The sharing of resources of a server constitute a service. Whether a computer is a client, a server, or both, is determined by the nature of the application that requires the service functions. For example, a single computer can run web server and file server software at the same time to serve different data to clients making different kinds of requests. Client software can also communicate with server software within the same computer. Communication between servers, such as to synchronize data, is sometimes called [inter-server](https://en.wikipedia.org/wiki/Inter-server) or server-to-server communication.

**7.3.2 OPTIMAL KEY MANAGEMENT:**

The EPS supports two types of handovers that are referred to as intra- and inter-MME handovers, with the names reflecting the anchor points involved. In the intra-MME handover, preparation for it occurs between the source and target eNodeBs in the same MME through a direct interface between base stations. In contrast, in the inter-MME handover, the preparation occurs via the MME without any direct signaling between base stations. As an alternative to the inter-MME handover, the UE and the MME may decide to run the full EPS-AKA to generate all security contexts from scratch. This alternative is more common in the inter-MME handover for security reasons. If different providers operate the two MMEs, the link between them is far from secure.

In this paper, we only consider the intra-MME handover in discussing the security weakness of key management in the handover because any security risks related to the inter-MME handover can be eliminated by running the full EPS-AKA. For efficiency, source eNodeB provides the next KeNB (K\*eNB) to the target network for use after the handover. Before the next EPS-AKA, a set of KeNB are linked to each other in what is known as handover key chaining to achieve backward key separation, source eNodeB generates the next KeNB from the current one by applying a one-way hash. To ensure forward key separation, the source eNodeB must capitalize on fresh keying material from an MME. An MME can provide fresh keying material to the target eNodeB only after the inter-eNodeB handover, and this fresh material is to be used in the next handover. The result is two-hop forward key separation in which the source eNodeB does not know the target eNodeB key only after two inter-eNodeB handovers. Handover key chaining includes two additional parameters as fresh keying material; these two are the Next Hop (NH) key and the NH Chaining Counter (NCC). An MME recursively generates a new NH key derived from KASME for each handover. NCC is a counter value for the NH key.

**7.3.3 AUTHENTICATION KEY AGREEMENT:**

AKA in 3GPP mobile networks have been increasing the possibility of rogue base station (i.e., false base station) attacks in the Global System for Mobile Communications (GSM); these attacks took the form of call stealing on unencrypted networks and call spoofing pointed out that the UMTS security displays vulnerabilities to a variant of rogue base station attacks. To the best of our knowledge, no serious rogue base station attacks on the EPS architecture have been reported in the public literature. Only the 3GPP standard has discussed theoretical rogue base station attacks. A few researchers initially surveyed EPS security. The authors in and provided a tutorial overview of EPS security, including the EPS-AKA and key management into handover key chaining and explored the operation of vertical and horizontal key derivation. The potential for DoS attacks on a specific UE by using radio signals was discussed . Recently, Koien pointed out that the delegation from the authentication server requires strong trust assumptions, which seems outdated in the LTE heterogeneous networks. He presented a mutual authentication directly between the user and the authentication server in online.

**7.3.4 LONG-TERM EVOLUTION SECURITY:**

Long-Term Evolution (LTE) is an emerging radio access network technology standardized in 3GPP and it is evolving as an evolution of Universal Mobile Telecommunications System (UMTS).It aims to provide seamless Internet Protocol (IP) connectivity between user equipments (UE) and the packet data network (PDN) without any disruption to the end users’ applications during mobility. The system is named evolved packet system (EPS) with two parts:

* System architecture evolution (SAE)
* Evolved packet core (EPC) network

There are five security levels:

* **Network access security (I):**
  + The set of security features that provides the UEs with secure access to the EPC and protect against various attacks on the radio link.
* **Network domain security (II):**
  + The set of security features that protects against attacks on the wire line network and enable nodes to exchange signaling data and user data in a secure manner
* **User domain security (III):**
  + The set of security features that provides a mutual authentication between the USIM and the ME before the USIM access to the ME.
* **Application domain security (IV):**
  + The set of security features that enables applications in the UE and in the provider domain to securely exchange messages.
* **Non 3GPP domain security (V):**
  + The set of features that enables the UEs to securely access to the EPC via non-3GPP access networks and provides security protection on the radio access link.

1. **SYSTEM STUDY**

**8.1 FEASIBILITY STUDY:**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**8.1.1 ECONOMICAL FEASIBILITY:**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

**8.1.2 TECHNICAL FEASIBILITY:**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**8.1.3 SOCIAL FEASIBILITY:**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

1. **SAMPLE SOURCE CODE**

**9.1 KEY GENERATOR:**

package com.smock.Server;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.ArrayList;

import javax.swing.ImageIcon;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JTextField;

import com.smock.util.AvailableNodes;

import com.smock.util.CodeGenerator;

import com.smock.util.ErrorManager;

public class Server {

private JFrame frame;

private JLabel senderL, priKeyL, pubKeyL,image,images;

private JTextField senderT, priKeyT, pubKeyT;

private Thread ser, serChk;

private ArrayList<String> availNodes;

private String netView[];

private boolean itrFlg=true;

public Server() {

frame = new JFrame("KeyGenarator");

frame.setLayout(null);

image = new JLabel();

image.setIcon(new ImageIcon("images\\2.jpg"));

image.setBounds(390,10, 251,234);

frame.add(image);

senderL = new JLabel("User Identified");

senderL.setBounds(5, 15, 150,30);

frame.add(senderL);

senderT = new JTextField();

senderT.setEditable(false);

senderT.setBounds(150, 15, 200, 30);

frame.add(senderT);

priKeyL = new JLabel("Public Key");

priKeyL.setBounds(5, 95, 150, 30);

frame.add(priKeyL);

priKeyT = new JTextField();

priKeyT.setEditable(false);

priKeyT.setBounds(150, 95, 200, 30);

frame.add(priKeyT);

pubKeyL = new JLabel("Private Key");

pubKeyL.setBounds(5, 175, 150, 30);

frame.add(pubKeyL);

pubKeyT = new JTextField();

pubKeyT.setEditable(false);

pubKeyT.setBounds(150, 175, 200, 30);

frame.add(pubKeyT);

images = new JLabel();

images.setIcon(new ImageIcon("images\\greenbackground.png"));

images.setBounds(0,0,690,350);

frame.add(images);

frame.setSize(650, 290);

frame.setLocation(600, 100);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setVisible(true);

//frame.setVisible(false);

checkServer();

serChk.start();

responseKey();

ser.start();

}

public void responseKey()

{

ser = new Thread(new Runnable()

{

public void run()

{

try

{

do

{

AvailableNodes nodes = new AvailableNodes();

availNodes = nodes.addAvailNode()

netView = (String[]) availNodes.toArray(new String[availNodes.size()]);

String whoIs;

ServerSocket rcvSkt1 = new ServerSocket(6655);

Socket skt1 = rcvSkt1.accept();

ObjectInputStream rcvObj1 = new ObjectInputStream(skt1.getInputStream());

whoIs = (String) rcvObj1.readObject();

senderT.setText(whoIs);

for (int chkWho = 0; chkWho < netView.length; chkWho++)

{

if (whoIs.equalsIgnoreCase(netView[chkWho]))

{

itrFlg = false;

break;

}

}

if(itrFlg==false)

{

CodeGenerator get = new CodeGenerator();

String pubKey = get.codeCreate();

pubKeyT.setText(pubKey);

String priKey = get.codeCreate();

priKeyT.setText(priKey);

sendKey(whoIs, pubKey, priKey);

}

else

{

new ErrorManager("Found Intruder : You dont have rights to communicate");

}

itrFlg = true;

skt1.close();

rcvSkt1.close();

} while (true);

}

catch (Exception e)

{

new ErrorManager("Server : "+e.toString());

e.printStackTrace();

}

}

});

}

public void sendKey(String destination, String puKey, String prKey)

{

try

{

Socket sendSkt1 = new Socket(destination, 5501);

ObjectOutputStream sendObj1 = new ObjectOutputStream(sendSkt1.getOutputStream());

sendObj1.flush();

String packMsg = puKey + "|" + prKey ;

sendObj1.writeObject(packMsg);

sendObj1.close();

sendSkt1.close();

}

catch (Exception e)

{

new ErrorManager("Server : "+e.toString());

e.printStackTrace();

}

}

public void checkServer()

{

serChk = new Thread(new Runnable() {

public void run()

{

try

{

do

{

ServerSocket rcvSkt1 = new ServerSocket(4444);

Socket skt1 = rcvSkt1.accept();

skt1.close();

rcvSkt1.close();

} while (true);

}

catch (Exception e)

{

new ErrorManager("Server : "+e.toString());

e.printStackTrace();

}

}

});

}

}

**9.2 AVAILABLE NODES:**

package com.smock.util;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayList;

public class AvailableNodes

{

public String command;

public Process p2;

public BufferedReader input;

public String result;

public ArrayList<String> listNode;

public StringBuffer strbuf;

public int cnt = 1;

public int cntLen = 0;

public ArrayList<String> addAvailNode()

{

command = "net view";

try

{

p2 = Runtime.getRuntime().exec(command);

input = new BufferedReader(new InputStreamReader(p2.getInputStream()));

listNode = new ArrayList<String>();

strbuf = new StringBuffer();

while ((result = input.readLine()) != null) {

cnt++;

if (cnt > 4)

{

strbuf.append(result);

if (strbuf.charAt(0) == '\\')

{

for (int i = 0; i < strbuf.capacity(); i++)

{

if (strbuf.charAt(i) == (' '))

{

cntLen = i;

break;

}

}

String temp = strbuf.substring(2, cntLen);

listNode.add(temp);

strbuf.delete(0, strbuf.length());

}

}

}

}

catch (Exception e)

{

System.out.println("Error " + e);

}

return listNode;

}

public static void main(String args[])

{

AvailableNodes s=new AvailableNodes();

ArrayList<String> array=new ArrayList<String>();

array=s.addAvailNode();

System.out.println(array);

}

}

**9.3 ENCRYPT/DECRYPT:**

package com.smock.util;

import java.security.InvalidKeyException;

import java.security.Key;

import javax.crypto.BadPaddingException;

import javax.crypto.Cipher;

import javax.crypto.IllegalBlockSizeException;

import javax.crypto.KeyGenerator;

public class EncryptDecrypt

{

private static String algorithm = "AES";

private static Key key = null;

private static Cipher cipher = null;

//private String input=null;

public String EncryptDecrypt1(String message)throws Exception

{

setUp();

byte[] encryptionBytes = null;

String input = message;

encryptionBytes = encrypt(input);

String str=encryptionBytes.toString();

// System.out.println(encryptionBytes);

//System.out.println("Recovered: " + decrypt(encryptionBytes));

String str2=str+"|"+decrypt(encryptionBytes);

return str2;

}

private static byte[] encrypt(String input)throws InvalidKeyException,BadPaddingException,IllegalBlockSizeException {

cipher.init(Cipher.ENCRYPT\_MODE, key);

byte[] inputBytes = input.getBytes();

return cipher.doFinal(inputBytes);

}

private static String decrypt(byte[] encryptionBytes)throws InvalidKeyException,BadPaddingException,IllegalBlockSizeException {

cipher.init(Cipher.DECRYPT\_MODE, key);

byte[] recoveredBytes =

cipher.doFinal(encryptionBytes);

String recovered =

new String(recoveredBytes);

return recovered;

}

private static void setUp() throws Exception

{

key = KeyGenerator.getInstance(algorithm).generateKey();

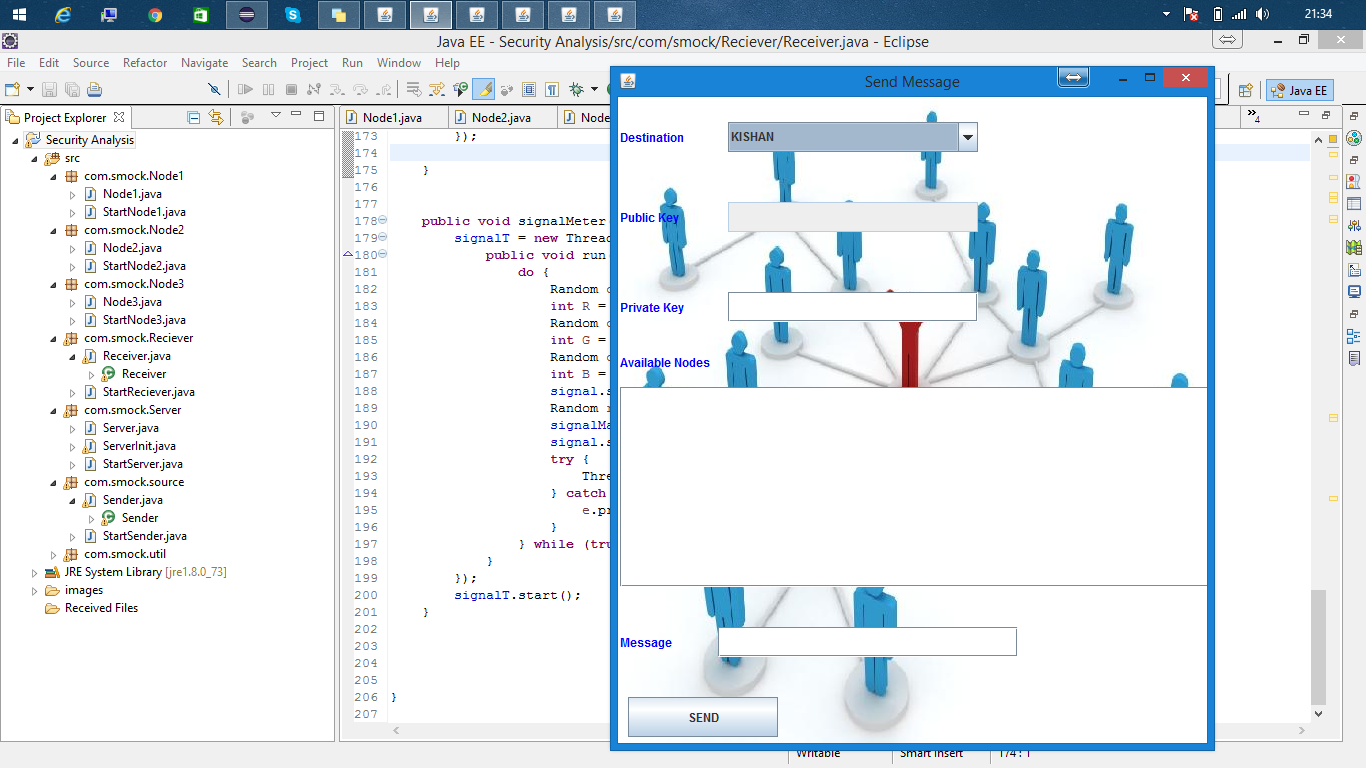
cipher = Cipher.getInstance(algorithm);

}

}

**10. SCREENSHOTS**

**10.1 Sender:**

****

**Fig 10.1: Senders side of the message transfer**

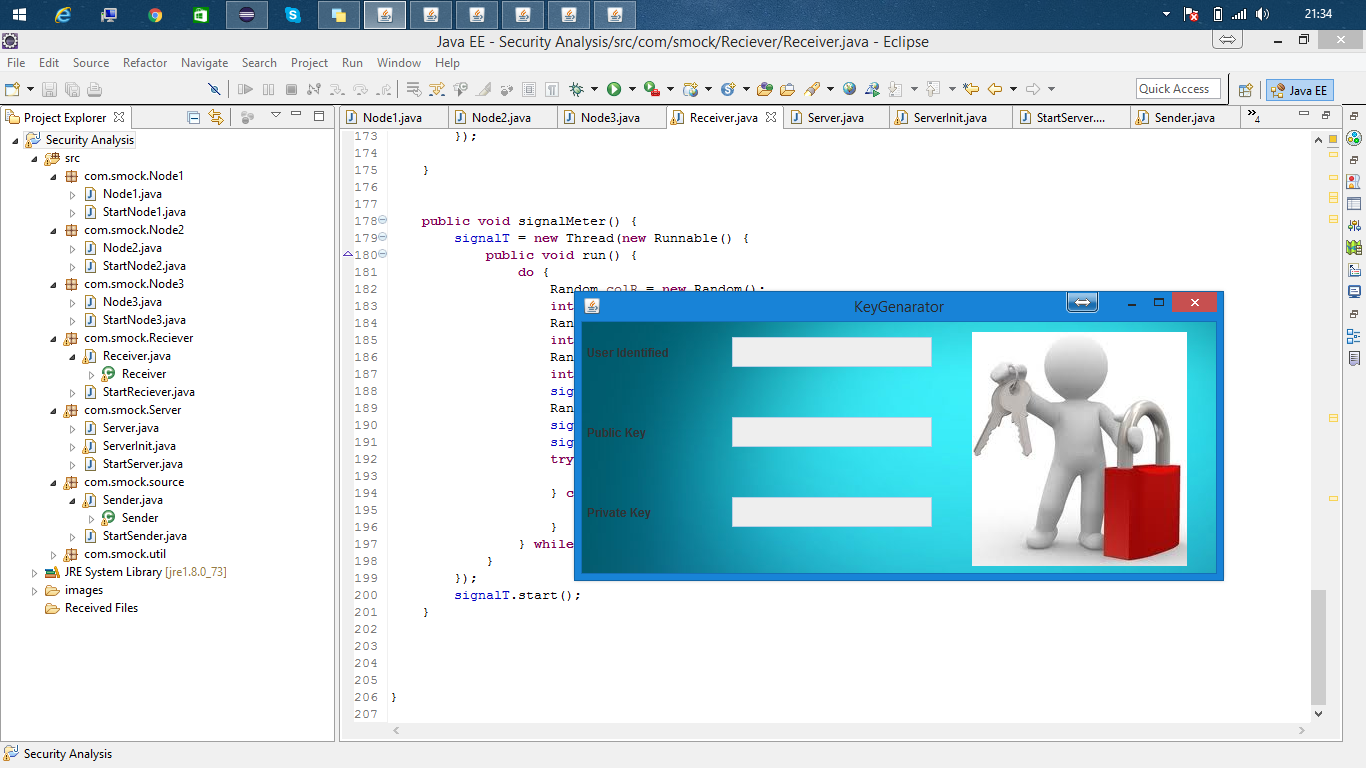
Destination - the sender can select a node to which he can send the message from a list of the available nodes in the wireless network.

Once the destination is selected the sender calls the key generator which provides the public and the private key that will be used in the message transfer and displays them in the respective fields.

Available nodes- This shows all the available nodes in the wireless network on scanning.

Message: Here the sender enters the message that has to be sent to the receiver.

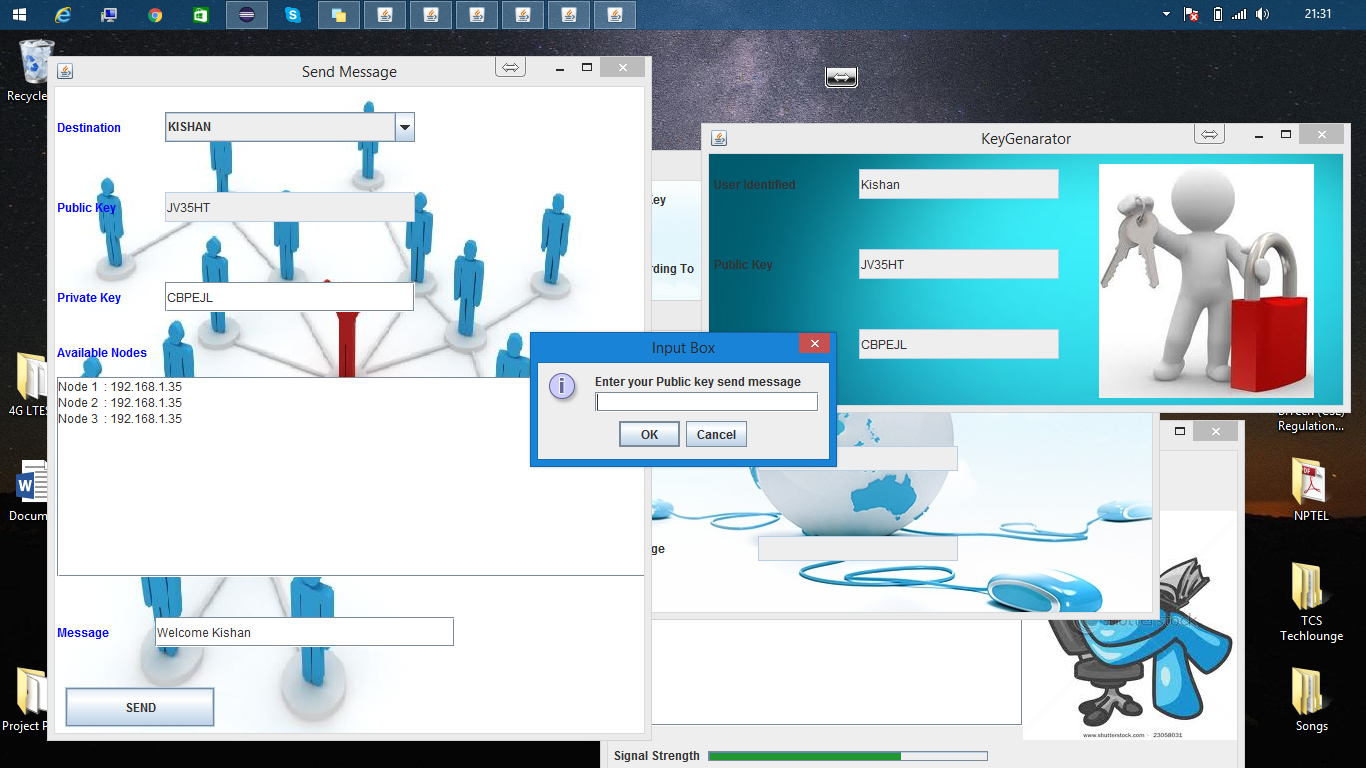
**10.2 Key Generator:**

****

**Fig 10.2: The Key generator**

The Key Generator is the heart of the code. It provides the sender with the private and the public key and also provides hash keys for the all the nodes present in the message transfer path with a hash key using a one way hash function to provide backward key separation.

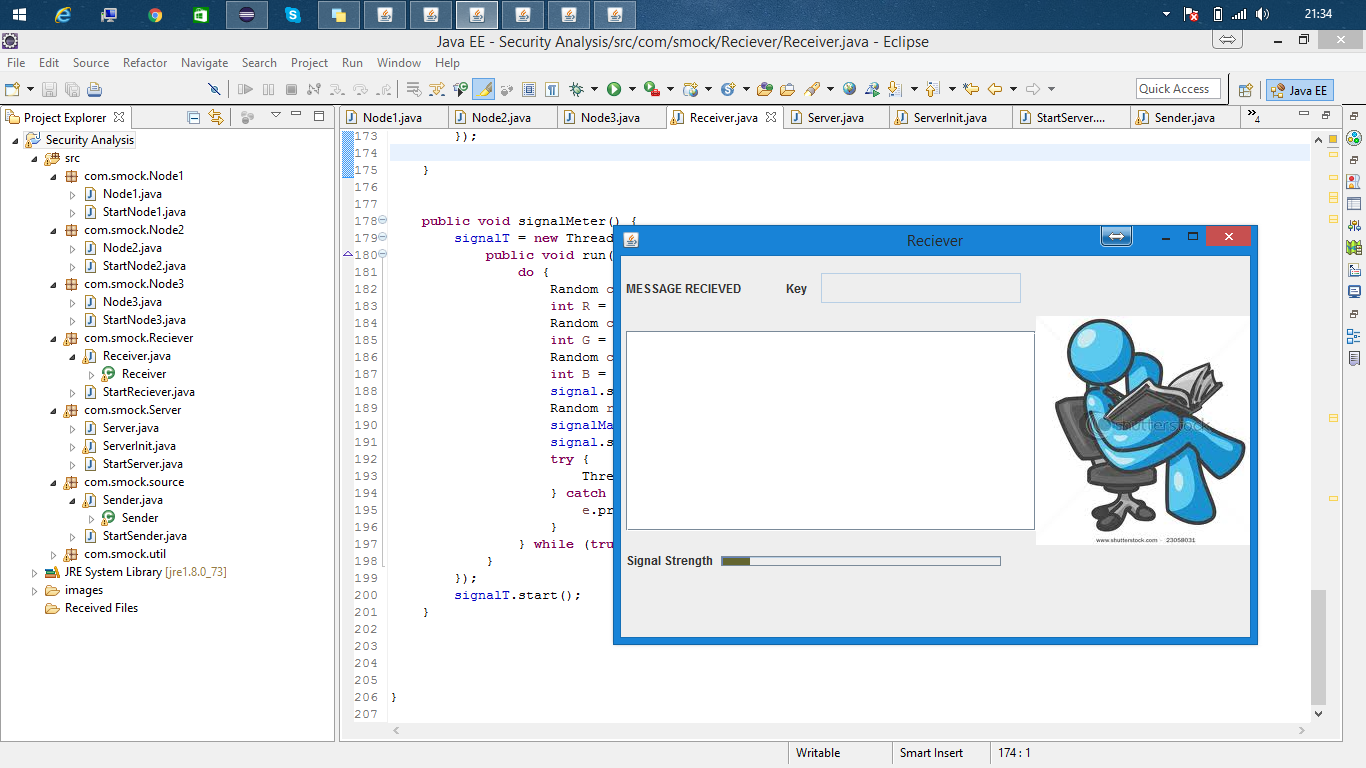
**10.3 Step1**

****

**Fig 10.3 The first step in the data transfer**

Once we have selected the destination , the sender calls the key generator which in turn provides the sender with the private and the public key . Then we get a “scan nodes” button which gives all the available nodes in the network. After entering the message in the message field and clicking on “send” a pop up box appears asking the user to enter the public key to encrypt the message and send it to the next hop.

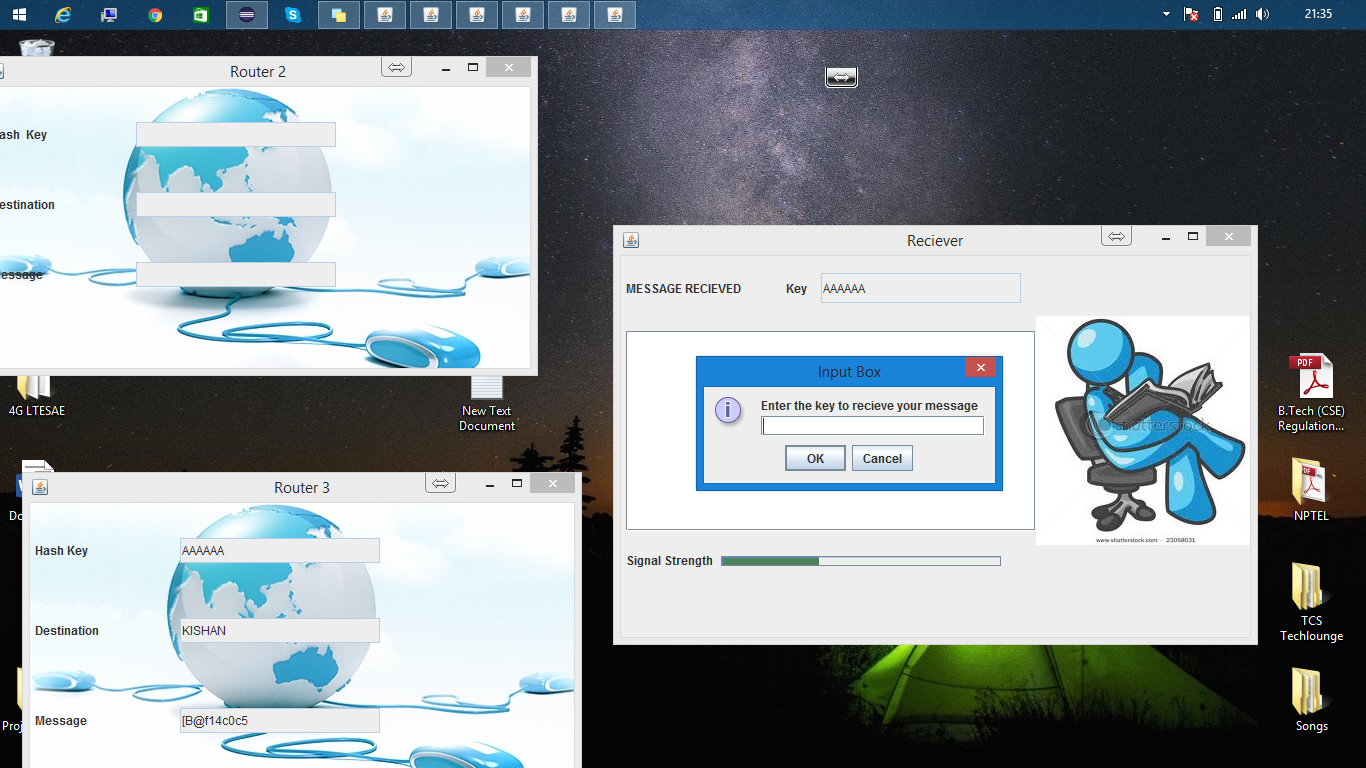
**10.4 Receiver:**

****

**Fig 10.4: The receivers side of the data transfer**

On the receivers side a key has to be entered which is provided by the key generator, on receiving the message in order to decrypt and print the message sent by the sender on the screen. Any mistake in the key , will not print the message on the screen.

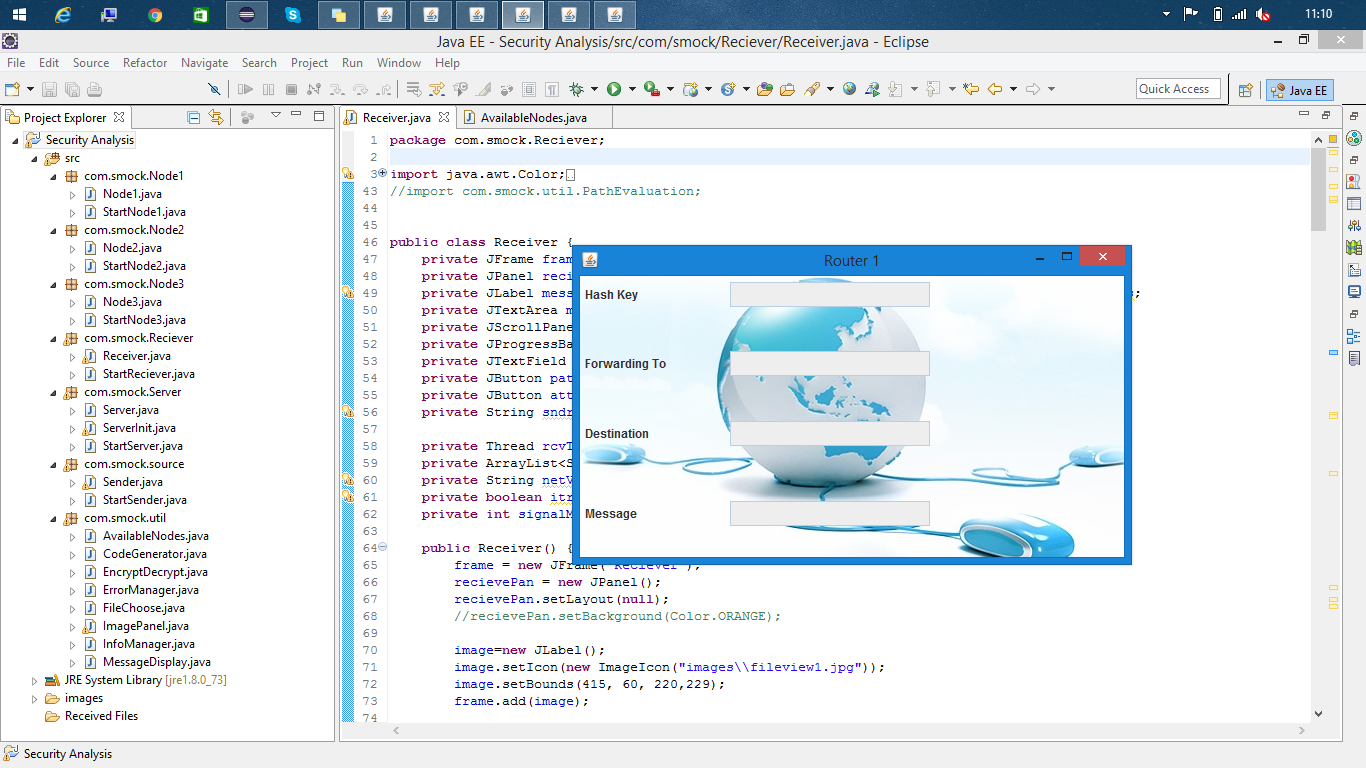
**10.5 Receivers End**

****

**Fig 10.5 Execution on the receivers end.**

On the receivers side a key has to be entered which is provided by the key generator by using one way hash function, on receiving the message in order to decrypt and print the message sent by the sender on the screen. Any mistake in the key , will not print the message on the screen.

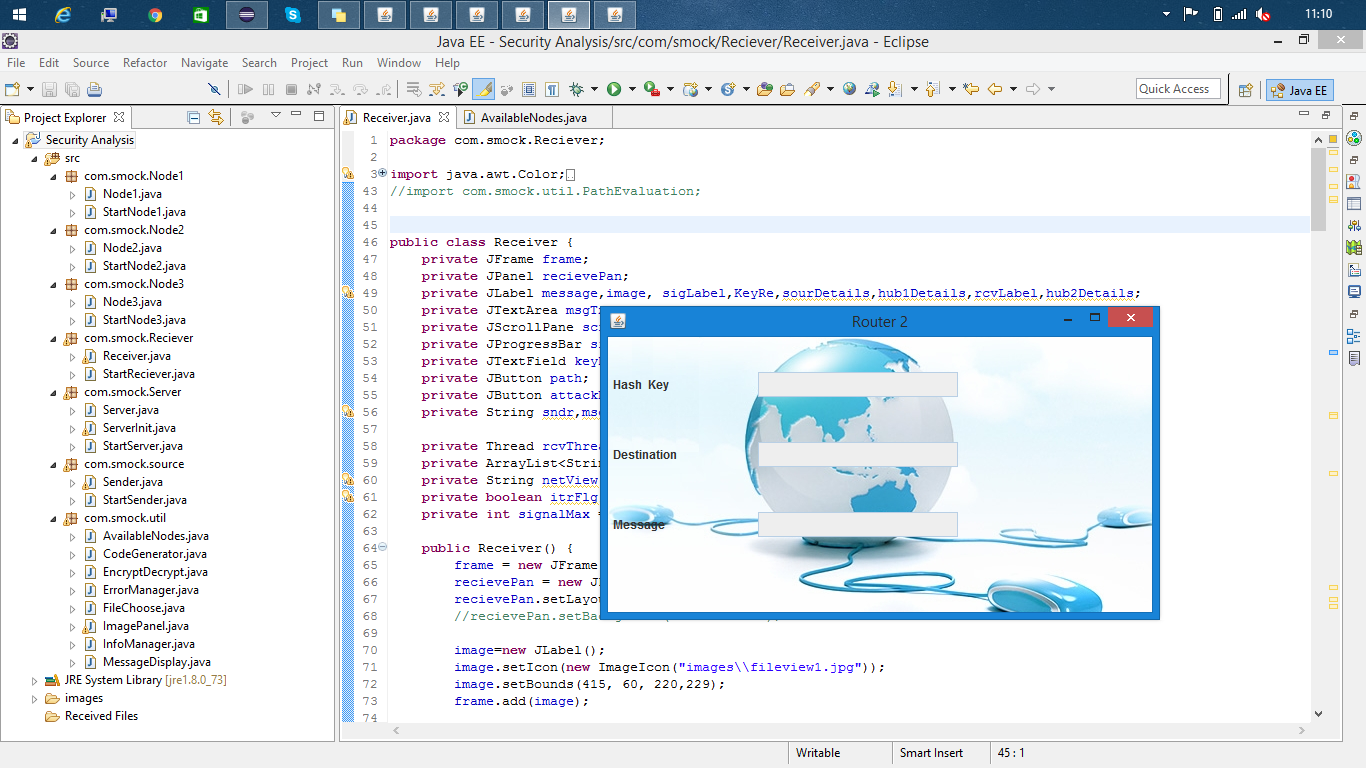
**10.6 Router 1:**

****

**Fig 10.6: Router 1 screen which shows the next hop, destination , hash key and the encrypted message.**

Router 1 receives the message from the sender which it encrypts using the hash key provided to it by the key generator. The screen shows the immediate next hop of the message and also the final destination of the message.

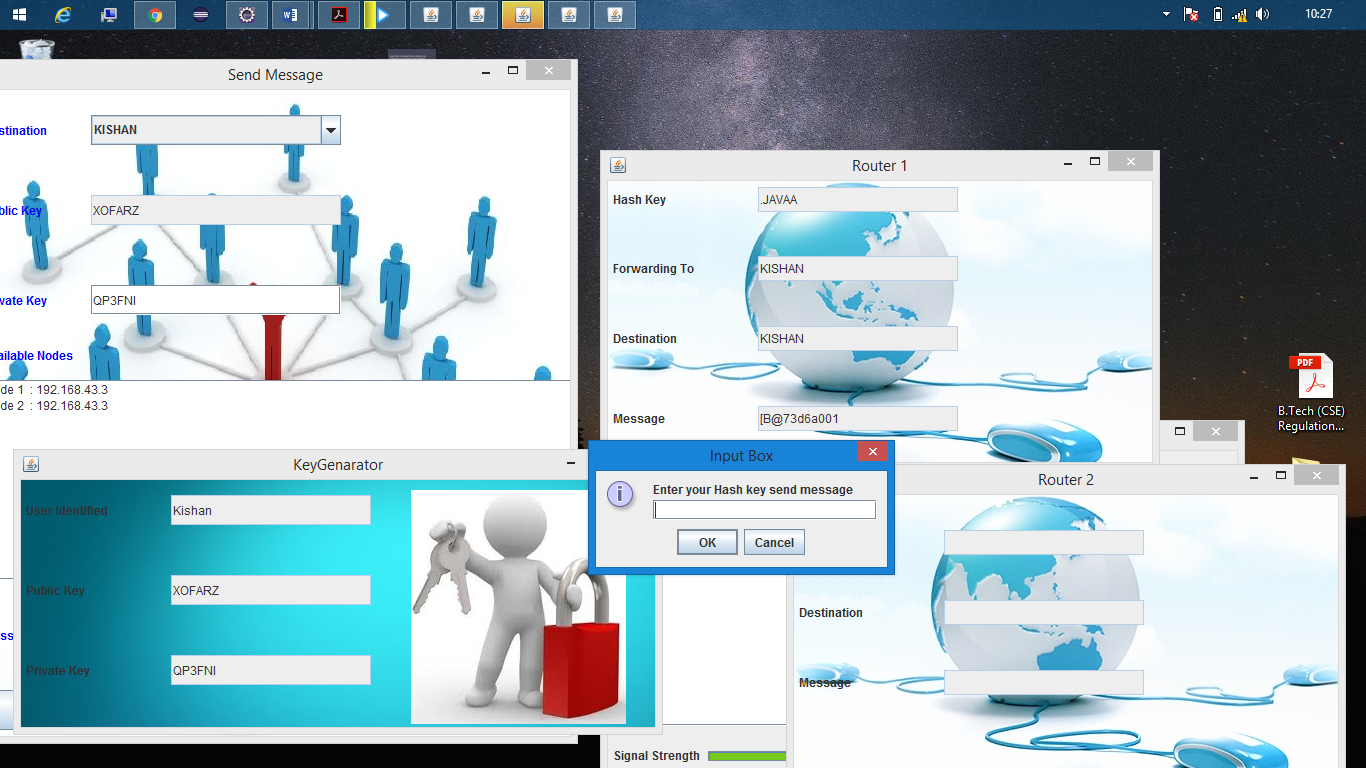
**10.7 Router 2:**

****

**Fig 10.7: Router 2 screen which shows the destination , hash key and the encrypted message.**

Router 2 receives the message from the Router 1 which it encrypts using the hash key provided to it by the key generator. The screen shows the final destination of the message and the has key used for the encrypting the message it receives from the previous router.

**10.8 Key generation at the nodes**

****

**Fig 10.8 Key generation at the nodes**

The nodes receive a hash key from the key generator , which uses a one way has function to generate the key. This key must be entered in order to encrypt the already encrypted data and send it to the next hop.

**11. SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**11.1 TYPES OF TESTS:**

Unit testing: Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing: Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test: Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked

System Test: System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points

*W*hite Box Testing: White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing: Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Test strategy and approach: Field testing will be performed manually and functional tests will be written in detail.

Test objectives:

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

Features to be tested:

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**11.2 TEST CASES:-**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Test Description | Steps | Expected | Actual | Status |
| 1. | At Source | Enter Random public key. | Sorry Your key is not valid | Same as expected. | Pass |
| 2. | At Source | Enter valid public key. | Message reaches the node. | Same as expected. | Pass |
| 3. | At Node | Enter invalid hash key. | Invalid Key | Same as expected. | Pass |
| 4. | At Source | Select a destination | Should give the public and private key. | Found Intruder. Rights not there to access. | Fail |
| 5. | At Source | Select a destination | Should give the public and private key. | Please try again later receive key. | Fail |
| 6. | At Node | Enter correct hash key. | Message passes to the next node. | Same as expected. | Pass |
| 7. | At Receiver | Enter valid hash key. | Display message | Same as expected. | Pass |
| 8. | At Source | Scan Nodes | Should give the available nodes IP addresses. | Same as expected. | Pass |
| 9. | At Source | When we click Send button | Should display the public key entry box | Same as expected. | Pass |
| 10. | At Node | After we press OK for encryption | Should display the encrypted message | Same as expected | Pass |

**12.CONCLUSION**

LTE and LTE-A are exciting and important technologies for the future of communications. This is true not only in the case of mobility applications, but for special fixed broadband applications in the countryside, too. LTE is installed today in first locations and LTE-A is on its way regarding standardization. Technology for the user terminal, e.g., modems for the laptop or smart phones with LTE functionality is already available today. It is important to analyse the requirements of future access technologies already at an early stage, in order to optimize the underlying transport network architectures.

The proposed idea will protect the 4G network during the initial call setup phase, periodic time based key exchange to ensure the call security and the seed exchange for the other end integrity check. The attacker can launch a variety of active and passive attacks. Thus the key management solution best fits the 4G SAE / LTE architecture. At the same time this system is maintaining a multilayered and multidimensional security approach. The two aspects are the selection of a straight forward ciphering key hierarchy distribution mechanism and providing a layered network security architecture, novel solutions for tackling LTE/SAE security issues on 4G wireless networks. Light weight key management system is the best solution than other alternatives because it offers a fast and secure transmission by adding a minimum design overhead. The key sharing based on the architecture time is used to protect the voice calls 4G. Therefore, there is a significant need for secure key management between the two nodes. Key sharing rules will be shared between the call ends (both nodes to make a call) during the initial handshake. The original key will be obtained and matched for integrity. If the key matches, the data would be exchanged between the two nodes, if the call is terminated flashing message integrity violation on the end of the spammer. In the future, the comparative analysis can be performed with higher level of performance analysis using the higher number of parameters. Also, the techniques under the survey can be improved or mixed in order to improve the overall performance of the scheme.

**13. REFERENCES**

1. Jin Cao, maode ma, IEEE Hui li, Yueyu Zhang and Zhenxing luo: “A survey on security aspects for LTE and LTE-A networks”, IEEE Communications Surveys & Tutorials., VOL.16, May 2014.

2. Chan-kya Han and Hyoung-kee choi, “Security Analysis Of Handover Key Management In 4G LTE/SAE Networks”, IEEE Transactions on Mobile Computing ,VOL.13 , NO.2, FEB 2014.

3. Mohsen M.Tantaway, Adly S.Tag Eldein and Esraa Mosleh Eid:” Performance Analysis of Multicast Security in LTE”, International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), VOL.2, ISSUE 5, Sept-Oct 2013.

4. Anand R.Prasad and Xiaowei Zhang: “Overview of LTE/SAE Security”, IEEE Transactions on Smart Processing and Computing. vol.2, no.1, February 2013.

5. Y. Zheng, D. He, L. Xu, and X. Tang: “Security Scheme for 4G Wireless Systems” Proc. Communications, Circuits and Systems, pp. 397- 401, May 2005.

6. “Cryptography and Network Security” 'William Stallings, Pearson Education, 2007.