**A PROJECT REPORT ON**

**A New Architecture for Network Intrusion Detection and Prevention**

**Submitted to \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* University for the partial fulfillment of the Requirement for the**

**Award of Degree for**

**\*\*\*\*\*\*\*\*\*\*\*Course Name\*\*\*\*\*\*\*\*\*\*\*\*\***

**Done By**

**Mr. /Miss \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*Institute of Management and Computer Science**

**CERTIFICATE**

**This is to certify that Mr., /Miss \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* bearing Roll No. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* have developed software project Titled \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* For \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Software Solutions as Partial fulfillment for the award of the Degree of \*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Head of the Department Principal**

**\*\*\*\*\*\*College Name\*\*\*\*\*\***

**External**

**ACKNOWLEDGEMENT**

At every outset I express my gratitude to almighty lord for showering his grace and blessings upon me to complete this project.

Although our name appears on the cover of this book, many people had contributed in some form or the other form to this project Development. We could not done this project without the assistance or support of each of the following we thank you all.

I wish to place on my record my deep sense of gratitude to my project guide, **Mr. \*\*\*\*\*\*, \*\*\*\*\* Software Solutions,** for his constant motivation and valuable help through the project work. Express my gratitude to **Mr. \*\*\*\*\*\***, Director of \*\*\*\*\*\*\* **Institute of Management & Computer Sciences** for his valuable suggestions and advices throughout the \*\*\*\*\* course. I also extend my thanks to other Faculties for their Cooperation during my Course.

Finally I would like to thank my friends for their cooperation to complete this project.

\*\*\*\*\*\*\*Your Name\*\*\*\*\*\*\*\*\*

**Abstract:**

This paper presents an investigation, involving experiments, which shows that current network intrusion, detection, and prevention systems (NIDPSs) have several shortcomings in detecting or preventing rising unwanted traffic and have several threats in high-speed environments. It shows that the NIDPS performance can be weak in the face of high-speed and high-load malicious traffic in terms of packet drops, outstanding packets without analysis, and failing to detect/prevent unwanted traffic. A novel quality of service (QoS) architecture has been designed to increase the intrusion detection and prevention performance. Our research has proposed and evaluated a solution using a novel QoS configuration in a multi-layer switch to Organize packets/traffic and parallel techniques to increase the packet processing speed. The new architecture was tested under different traffic speeds, types, and tasks. The experimental results show that the architecture improves the network and security performance which is can cover up to 8 Gb/s with 0 packets dropped. This paper also shows that this number (8 GB/s) can be improved, but it depends on the system capacity which is always limited.

**LIST OF CONTENTS**

1. Introduction

1.1 Purpose

1.2 Scope and Objective

1.3 Motivation

2. Literature Survey

3. Fundamental Concepts on (Domain)

4. System Analysis

4.1 Modules Description

4.2 Feasibility Study

4.2.1 Economic Feasibility

4.2.2 Operational Feasibility

4.2.3 Technical Feasibility

5. System Requirements Specification

5.1 Introduction

5.2 Purpose

5.3 Functional Requirements

5.4 Non Functional Requirements

5.5 Input & Output Design

5.6 Hardware Requirements

5.7 Software Requirements

6. System Design

6.1 System Specifications

6.2 System Components

6.3 DFD’s

6.4 UML Diagrams ( 9 types )

6.5 Data Dictionaries and ER Diagram

7. Implementation

7.1 Technology Description

8. Coding

9. System Testing

9.1 Testing Methodologies

9.2 Test cases

10. Output Screens (FORMS & REPORTS)

11. Future Enhancement

12. Conclusion

13. References

14. Bibliography

1. **Introduction**

Information technology (IT) inﬂuences almost every aspect of modern life. Today, various devices are available to meet users’ requirements such as high machine processor speed, and fast networks. Alongside our increasing dependence on IT, there has unfortunately been a rise in security incidents. Threats and attacks may range from stealing personal infor- mation from a laptop or network server to stealing the most top-secret information stored on a Security Intelligence Ser- vice (SIS). Furthermore, hackers can snoop on users’ online purchases by eavesdropping on their credit card details, or, even more alarmingly, safety-critical systems can be com- promised. Multi-faceted attacks and threats have made the implementation of security systems more challenging. Hack-ers have evolved along with the sophistication of the IT industry. For example, hackers exploit the developments in computer processors and network speeds to increase the volume and speed of malicious traffic that might constitute a Denial of Service (DoS) or Distributed Denial of Ser- vice (DDoS) attack [1]–[3]. Network security is therefore extremely important and has developed into an industry aimed at improving applications and hardware platforms to identify and stop network threats. One of the most established concepts in information secu- rity is a defense-in-depth approach which utilizes a multi- layered structural design, in which ﬁrewalls, vulnerability assessment tools (anti-viruses and worms), and IDPS (Intru- sionDetectionandPreventionSystems)areemployedtopre- vent any hostile endeavours on network systems and servers. The Network Intrusion Detection and Prevention System (NIDPS) has been designed to serve as the last point of defenseinthenetworkarchitecture.NIDPSmonitorthetrans- portationofnetworktrafﬁcforanymaliciousanduncomfort- able activities and create alerts when operating in detection mode or block packet alerts when operating in prevention node

* 1. **Purpose:**

This paper presents an investigation, involving experiments, which shows that current network intrusion, detection, and prevention systems (NIDPSs) have several shortcomings in detecting or preventing rising unwanted traffic and have several threats in high-speed environments. It shows that the NIDPS performance can be weak in the face of high-speed and high-load malicious traffic in terms of packet drops, outstanding packets without analysis, and failing to detect/prevent unwanted traffic.

**Scope and Objective:**

A novel quality of service (QoS) architecture has been designed to increase the intrusion detection and prevention performance. Our research has proposed and evaluated a solution using a novel QoS configuration in a multi-layer switch to Organize packets/traffic and parallel techniques to increase the packet processing speed. The new architecture was tested under different traffic speeds, types, and tasks. The experimental results show that the architecture improves the network and security performance which is can cover up to 8 Gb/s with 0 packets dropped. This paper also shows that this number (8 GB/s) can be improved, but it depends on the system capacity which is always limited.

**2. Literature Survey:**

**DDoS Attack Protection in the Era of Cloud Computing and Software-Deﬁned Networking**

Cloud computing has become the real trend of enterprise IT service model that offers cost-effective and scalable processing. Meanwhile, Software-Deﬁned Networking (SDN) is gaining popularity in enterprise networks for ﬂexibility in network management service and reduced operational cost. There seems a trend for the two technologies to go hand-in-hand in providing an enterprise’s IT services. However, the new challenges brought by the marriage of cloud computing and SDN, particularly the implications on enterprise network secu- rity, have not been well understood. This paper sets to address this important problem. We start by examining the security impact, in particular, the impact on DDoS attack defense mechanisms, in an enterprise network where both technologies are adopted. We ﬁnd that SDN technology can actually help enterprises to defend against DDoS attacksif the defense architecture is designed prop- erly. To that end, we propose a DDoS attack mitigation architecture that integrates a highly programmable net- work monitoring to enable attack detection and a ﬂexible controlstructuretoallowfastandspeciﬁcattackreaction. The simulation results show that our architecture can effectively and efﬁciently address the security challenges brought by the new network paradigm. As cloud computing provides on-demand, elastic, and accessible computing services, more and more enterprises begin to embrace this paradigm shift by moving their database and applications into the cloud. At the same time, another epochal concept of the Internet architecture comes to forefront, i.e., Software- Deﬁned Networking (SDN). While cloud computing facilitates the management of computation and storage resources, SDN is proposed to address another labori- ous issue hindering the evolvement of today’s Internet, i.e., the complicated network management. Besides the fact that SDN has been proposed as a candidate of the next generation Internet architecture, companies like Google have already adopted SDN in their internal data centers. Thus, the arrival of the era when cloud computing and SDN go hand-in-hand in providing enterprise IT services is looming on the horizon. Besides all the widely perceived beneﬁts, the mar- riage between cloud computing and SDN may also introduce potential risks, especially on network secu- rity. Among all the network security problems, we ﬁrst take a look at Denial-of-Service (DoS) attack. A DoS attack and its distributed version, Distributed Denial-of-Service (DDoS) attack, attempt to make a service unavailable to its intended users by draining thesystem or network resource. Although network security experts have been devoting great efforts for decades to address this issue, DDoS attacks continue to grow in frequency and have more impact recently. Existing DDoS attack defense solutions (to list a few [1], [2], [3], [4]) assume a fully controlled network by the network administrators of enterprises. Therefore, the network administrators could place certain hardware pieces in the network to detect or mitigate DDoS at- tacks. However, in the new network paradigm of cloud computing and SDN, these assumptions no longer stand. Other researchers [5], [6] focus on exploiting the beneﬁts of cloud or SDN to defend DDoS attacks. But their target victims still reside in the traditional network environment, which makes their solutions un- suitable for the new network paradigm. To the best of our knowledge, little effort in research community has been made to look into the potential problems or opportunities to defend DDoS attacks in the new enterprise network environment that adopts both cloud computing and SDN.

**Distributed Denial of Service (DDOS) Attacks in Cloud Computing**

Cloud computing is the use of computing resources (hardware & software) that delivered as service over internet. For sensitivity or security of data, existing solutions usually apply cryptographic methods by using encryption and decryption keys and giving these keys to only authorized users. But when we apply these methods to real cloud the problem of simultaneously achieving fine-grainedness, scalability and data confidentiality of access control actually still remains unsolved. In the cloud computing, the prevalence and sophistication of DoS and DDoS on the internet are rapidly increasing. Service providers are under mounting pressure to prevent, monitor and mitigate DoS/DDoS attacks directed towards their customers. Attacks that are seen every day on the internet in the cloud computing include Zombie attack, phishing attack, DoS and DDoS attack, man-in-middle attack, service injection attack, metadata spoofing attack. These attacks can cause damage and wide spread out gages when directed at a service provider’s infrastructure. The monitoring and mitigation of these attacks is a crucial part of a service providers operation. In this paper we have studied cloud computing, attacks (mainly DDoS attacks) on cloud computing and techniques to cover these attacks. Further we have tried to explain the pros and cons of different techniques and its impact on real world cloud. In cloud computing, the word cloud represents the metaphor “the internet” and the phrase cloud computing “means a type of internet based computing” where different services such as servers, storage and application are delivered to an organizations, computers and devices through the internet. Cloud computing has emerged as a way for IT businesses to increase capabilities on the fly without investing much in new infrastructure, training of personals or licensing new software [1]. National Institute of Standards and Technology NIST [2] defines Cloud computing as a “model for enabling ubiquitous, convenient, on demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and delivered with minimal managerial effort or service provider interaction”.

The Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are the most common but fatal type of attack on cloud service providers (CSPs) which are working hard to prevent, monitor and mitigate these types of attacks as the frequency of these types of attacks have risen sharply in the last few years. DDoS are directed at service provider’s infrastructure can be very damaging. In cloud computing, the DoS or DDoS attack is when a machine or network resources unavailable to its intended users. DDoS attacks are sent by two or more persons or bots. DDoS attacks are sent by one person or system. In this paper we have discussed the most common types of DoS/DDoS attacks seen on the internet and ways that service providers can prevent or mitigate damages from the attack threats. The monitoring of DoS/DDoS and black hole filtering became mandatory as entry for service providers to sell the service of internet in the financial industry. The financial industry is easily susceptible to DoS/DDoS attacks as millions of consumers move to electronic bill payments, purchases and On-line banking.

**Intrusion Detection System in Cloud Computing**

Today, Cloud Computing is the preferred choice of every IT organization since it provides flexible and pay-per- use based services to its users. However, the security and privacy is a major hurdle in its success because of its open and distributed architecture that is vulnerable to intruders. Intrusion Detection System (IDS) is the most commonly used mechanism to detect attacks on cloud. This paper provides an overview of different intrusions in cloud. Then, we analyze some existing cloud based intrusion detection systems (IDS) with respect to their type, positioning, detection time, detection technique, data source and attacks they can detect. The analysis also provides limitations of each technique to evaluate whether they fulfill the security requirements of cloud computing environment or not. We emphasize the deployment of IDS that uses multiple detection methods to cope with security challenges in cloud. Cloud Computing offers omnipresent, convenient, demand-based access to a shared group of configurable computing resources (like storage, network, services applications and servers) that can be quickly provisioned and released with least management effort or service provider interactions [1]. It provides services to its users in different ways: Infrastructure as a Service (IaaS), where the user has control over complete virtual machines [2] such as Eucalyptus, Open Nebula [3]. Platform as a Service (PaaS), where the user can deploy user-created applications in cloud if the provider supports the languages, APIs, and tools used for creating application, [1] like Google App Engine, Microsoft’s Azure [3]. Software as a Service (SaaS) which enables users to execute provider’s applications [1] such as Google apps [3]. These services are provided via the Internet. There are four deployment models for cloud: Public cloud, its infrastructure is intended to be used by general public and managed by a governmental academic or business organization. Private cloud, it is deployed for a particular organization having

multiple users. Its management is the responsibility of organization using its services or a third party. Community cloud which is deployed for use by a particular group of users from organizations having common goals. It can be managed by any of the organizations within that group or a third party. Hybrid cloud, its infrastructure consists of two or more different cloud infrastructures (public, private, or community) that ensures the portability of applications and data using a standard technology. The unique features of clouds forming hybrid cloud are retained [1].

**Improving Network Intrusion Detection System Performance through Quality of ServiceConfiguration and Parallel Technology**

This paper outlines an innovative software development that utilizes Quality of Service (QoS) and parallel technologies in Cisco Catalyst Switches to increase the analytical performance of a Network Intrusion Detection System (NIDS) when deployed in high-speed networks. We have designed a real network to present experiments that use a Snort NIDS to demonstrate the weaknesses of NIDSs, such as inability to process multiple packets and propensity to drop packets in heavy traffic and high-speed networks without analysing them. We tested Snort’s analysis performance, gauging the number of packets sent, analysed, dropped, filtered, injected, and outstanding. We suggest using QoS configuration technologies in a Cisco Catalyst 3560 Series Switch and parallel Snort NIDSs to improve NIDS performance and to reduce the number of dropped packets. Our results show that our novel configuration improves performance. In order to provide new developments and the highest- quality services, companies implement the latest technologies in their infrastructure. A company’s network plays a vital role in its business projects; it can achieve success in its business career by keeping its computer network up-to-date with the latest software and security techniques. Reliability and safety are the major concerns in enabling a company to achieve success and boost its progress. However, these networks can also be considered a major risk in any business project. Security issues have increased as technology has advanced. Fuchsberger [1] reported that, according to a survey conducted by Federal Bureau of investigation and Crime Scene of investigation (FBI/CSI), viruses are behind many attacks on business networks. Moreover, denial of service (DoS) attacks and unauthorized user access (which can be initiated from external or internal LAN sources) have also increased dramatically. In order to provide new developments and the highest- quality services, companies implement the latest technologies in their infrastructure. A company’s network plays a vital role in its business projects; it can achieve success in its business career by keeping its computer network up-to-date with the latest software and security techniques. Reliability and safety are the major concerns in enabling a company to achieve success and boost its progress. However, these networks can also be considered a major risk in any business project. Security issues have increased as technology has advanced. Fuchsberger [1] reported that, according to a survey conducted by Federal Bureau of investigation and Crime Scene of investigation (FBI/CSI), viruses are behind many attacks on business networks. Moreover, denial of service (DoS) attacks and unauthorized user access (which can be initiated from external or internal LAN sources) have also increased dramatically.

**Managing NFV using SDN and Control Theory**

Control theory and SDN (Software Deﬁned Net- working) are key components for NFV (Network Function Virtualization) deployment. However little has been done to use a control-theoretic approach for SDN and NFV management. In this demo, we describe a use case for NFV management using control theory and SDN. We use the management architecture of RINA (a clean-slate Recursive InterNetwork Architecture) to manage Virtual Network Function (VNF) instances over the GENI testbed. We deploy Snort, an Intrusion Detection System (IDS) as the VNF. Our network topology has source and destination hosts, multiple IDSes, an Open vSwitch (OVS) and an OpenFlow controller. A distributed management application running on RINA measures the state of the VNF instances and communicates this information to a Proportional Integral (PI) controller, which then provides load balancing information to the OpenFlow controller. The latter controller in turn updates trafﬁc ﬂow forwarding rules on the OVS switch, thus balancing load across the VNF instances. This demo demonstrates the beneﬁts of using such a control- theoretic load balancing approach and the RINA management architecture in virtualized environments for NFV management. It also illustrates that the GENI testbed can easily support a wide range of SDN and NFV related experiments. NFV elastic management includes tasks related to Virtual Network Function (VNF) stateful migration from one Virtual Machine (VM) to another, and adding or removing VNF instances depending on the load on the system [1], [2], [3]. NFV elastic management has recently received considerable attention in the research community [1], [2], [3]. However, most of this work focuses on VNF stateful migration. In this demo, we use a new internet architecture – the Recursive InterNetwork Architecture (RINA) [4] – to share VNF state information across the system and use a control-theoretic approach for managing load across VNF instances. To the best of our knowledge, this is the ﬁrst work that uses a control- theoretic approach to NFV management. Figure 1 shows an overview of the system. We deploy Snort [5], an Intrusion Detection System (IDS) as the VNF. There can be multiple source and destination hosts and all traf- ﬁc directed from any source to any destination passes through Snort-IDS. VNF hosts run a distributed monitoring application (deployed over RINA) where each application instance shares the state of the VNF (i.e., load information) with the central controller. The controller runs a control-theoretic Proportional Integral (PI) control algorithm that balances load across the VNF instances by providing the OVS controller with load balancing information, which is then used to update the ﬂow forwarding rules on the OVS switch so new ﬂows are directed to less loaded VNF instances.

**3. Fundamental Concepts of Domain:**

3.1 Domain Fundamentals & Description

Introduction

This networking is primarily about TCP/IP network protocols and ethernet network architectures, but also briefly describes other protocol suites, network architectures, and other significant areas of networking. This networking tutorial is written for all audiences, even those with little or no networking experience

3.1 Networking Overview

It explains in simple terms the way networks are put together, and how data packages are sent between networks and subnets along with how data is routed to the internet. This networking tutorial is broken into five main areas which are:

1. Basics - Explains the protocols and how they work together
2. Media - Describes the cabling and various media used to send data between multiple points of a network.
3. Architecture - Describes some popular network architectures. A network architecture refers to the physical layout (topology) of a network along with the physical transmission media (Type of wire, wireless, etc) and the data access method (OSI Layer 2). Includes ethernet, Token Ring, ARCnet, AppleTalk, and FDDI. This main area of the networking tutorial can and should be skipped by those learning networking and read later.
4. Other Transport Protocols - Describes IPX/SPX, NetBEUI, and more.
5. Functions - Explains some of the functionality of networking such as routing, firewalls and DNS

The reader may read this networking tutorial in any order, but for beginners, it would be best to read through from the beginning with the exception of sections 2 (media), 3 (architecture), and 4 (other). At some point, however, the reader should be able to break from the basics and read about routing and IP masquerading. There are no links to various reading material or software packages inside this networking tutorial, except under the references section. This is because it is more structured, and makes it easier to keep the networking tutorial current.

This networking tutorial will first talk about the network basics so the reader can get a good grasp of networking concepts. This should help the reader understand how each network protocol is used to perform networking. The reader will be able to understand why each protocol is needed, how it is used, and what other protocols it relies upon. This networking tutorial explains the data encapsulation techniques in preparation for transport along with some of the network protocols such as IP, TCP, UDP, ICMP, and IGMP. It explains how ARP and RARP support networking. In functional areas, such as routers, several examples are given so the user can get a grasp on how networking is done in their particular situation. This networking tutorial covers routing, IP masquerading, and firewalls and gives some explanation of how they work, how they are set up, and how and why they are used. Firewalls and the available packages are described, but how to set them up is left to other documentation specific to the operating system and the package. Application protocols such as FTP and Telnet are also briefly described. Networking terms are also explained and defined.

This networking tutorial explains the setup of networking functions using Linux Redhat version 6.1 as an operating system (OS) platform. This will apply to server functions such as routing and IP masquerading. For more documentation on setting up packages, read documentation on this web site and other locations specific to the operating system and the package. If you know how to set up other operating servers such as Windows NT, you can apply the information in this networking tutorial to help you understand how to configure services on that OS platform.

This networking tutorial was written because I perceived a need for a basic networking document to explain how these networking services work and how to set them up, with examples. It will help a novice to learn networking more quickly by explaining the big picture concerning how the system works together. I have seen much good networking documentation, but little that explains the theory along with practical setup and applications.

A network consists of multiple computers connected using some type of interface, each having one or more interface devices such as a Network Interface Card (NIC) and/or a serial device for PPP networking. Each computer is supported by network software that provides the server or client functionality. The hardware used to transmit data across the network is called the media. It may include copper cable, fiber optic, or wireless transmission. The standard cabling used for the purposes of this document is 10Base-T category 5 ethernet cable. This is twisted copper cabling which appears at the surface to look similar to TV coaxial cable. It is terminated on each end by a connector that looks much like a phone connector. Its maximum segment length is 100 meters.

Network Categories

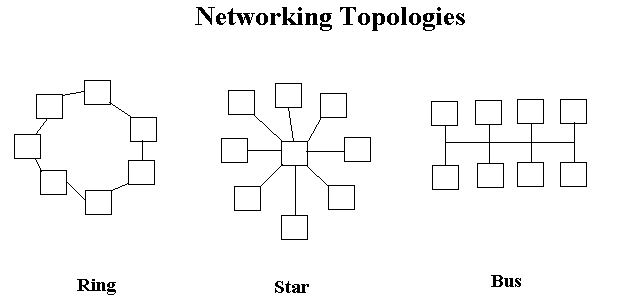
There are two main types of network categories which are:

* Server based
* Peer-to-peer

In a server based network, there are computers set up to be primary providers of services such as file service or mail service. The computers providing the service are are called servers and the computers that request and use the service are called client computers.

In a peer-to-peer network, various computers on the network can act both as clients and servers. For instance, many Microsoft Windows based computers will allow file and print sharing. These computers can act both as a client and a server and are also referred to as peers. Many networks are combination peer-to-peer and server based networks. The network operating system uses a network data protocol to communicate on the network to other computers. The network operating system supports the applications on that computer. A Network Operating System (NOS) includes Windows NT, Novell Netware, Linux, Unix and others.

|  |  |
| --- | --- |
| Three Network Topologies  The network topology describes the method used to do the physical wiring of the network. The main ones are bus, star, and ring. |  |



1. Bus - Both ends of the network must be terminated with a terminator. A barrel connector can be used to extend it.
2. Star - All devices revolve around a central hub, which is what controls the network communications, and can communicate with other hubs. Range limits are about 100 meters from the hub.
3. Ring - Devices are connected from one to another, as in a ring. A data token is used to grant permission for each computer to communicate.

There are also hybrid networks including a star-bus hybrid, star-ring network, and mesh networks with connections between various computers on the network. Mesh networks ideally allow each computer to have a direct connection to each of the other computers. The topology this documentation deals with most is star topology since that is what ethernet networks

1.3.2 Network Applications

There are three categories of applications with regard to networks:

1. Stand alone applications - Includes editors
2. Network versions of stand alone applications - May be licensed for multiple users.
3. Applications only for a network include databases, mail, group scheduling, groupware.

Models for network applications

1. Client-server - Processing is split between the client which interacts with the user and the server performing back end processing.
2. Shared file systems - The server is used for file storage and the processing of the file is done on the client computer.
3. Applications that are centralized - An example is a Telnet session. The data and the program run on the central computer and the user uses an interface such as the Telnet client or X server to send commands to the central computer and to see the results.

E-mail Systems

* Novell GroupWise - Also called Windows Messaging
* Microsoft Mail
* Microsoft Exchange - This is for the Microsoft Exchange Server. There is a Microsoft Exchange client for the Microsoft Exchange server and a client for an internet mail account only.
* Lotus Notes
* cc:Mail - From Lotus and IBM

Mail API

Mail application programming interfaces (APIs) allow e-mail support to be integrated into application programs.

* MAPI - Microsoft's Messaging API incorporated throughout Microsoft's office products provides support for mail at the application level.
* VIM - Vendor-Independent Messaging protocol from Lotus is supported by many vendors exclusive of Microsoft.

Message Handling Service (MHS)

* MHS and Global MHS by Novell
* MHS by OSI - It is called MOTIS (message-oriented text interchange system).

X.500

This is a recommendation outlining how an organization can share objects and names on a large network. It is hierarchical similar to DNS, defining domains consisting of organizations, divisions, departments, and workgroups. The domains provide information about the users and available resources on that domain, This X.500 system is like a directory. Its recommendation comes from the International Telegraph and Telephone Consultative Committee (CCITT).

**4. System Analysis:**

**4.1 Existing System:**

Threats and attacks may range from stealing personal information from a laptop or network server to stealing the most top-secret information stored on a Security Intelligence Service (SIS). Furthermore, hackers can snoop on users' online purchases by eavesdropping on their credit card details, or, even more alarmingly, safety-critical systems can be compromised. Multi-faceted attacks and threats have made the implementation of security systems more challenging. Hackers have evolved along with the sophistication of the IT industry. For example, hackers exploit the developments in computer processors and network speeds to increase the volume and speed of malicious traffic that might constitute the associate editor coordinating the review of this manuscript and approving it for publication was Ali Kashif Bashir. A Denial of Service (DoS) or Distributed Denial of Service (DDoS) attack. Network security is therefore extremely important and has developed into an industry aimed at improving applications and hardware platforms to

Identify and stop network threats.

**4.1.1 Drawbacks:**

One of the most established concepts in information security is a defense-in-depth approach which utilizes a multilayered structural design, in which firewalls, vulnerability assessment tools (anti-viruses and worms), and IDPS (Intrusion Detection and Prevention Systems) are employed to prevent any hostile endeavors on network systems and servers.

**4.1.2 Problem statement**

One of the most established concepts in information security is a defense-in-depth approach which utilizes a multilayered structural design, in which firewalls, vulnerability assessment tools (anti-viruses and worms), and IDPS (Intrusion Detection and Prevention Systems) are employed to prevent any hostile endeavors on network systems and servers.

**4.1.3 Proposed System**

One of the most established concepts in information security is a defense-in-depth approach which utilizes a multilayered structural design, in which firewalls, vulnerability assessment tools (anti-viruses and worms), and IDPS (Intrusion Detection and Prevention Systems) are employed to prevent any hostile endeavors on network systems and servers. The Network Intrusion Detection and Prevention System

(NIDPS) has been designed to serve as the last point of defense in the network architecture. NIDPS monitor the transportation of network traffic for any malicious and uncomfortable activities and create alerts when operating in detection mode or block packet alerts when operating in prevention node . The detection and prevention mechanisms of the NIDPS are grounded in observing the comparison of ingress packets (traffic) to any known attack through patterns (signature NIDPS mechanism) or identifying unknown malicious patterns from ingress traffic (anomaly NIDPS mechanism).

**4.1.4 Advantages**.

1. Counter intrusions or malicious attempts to access networks and systems.

2. Analyze network traffic and identify hackers' targets and techniques; and detect or prevent unwanted and malicious traffic.

3. The research investigates how QoS including DiffServ technology and parallelism can have impact in high-speed and heavy traffic networks using an industry standard switch and standard desktop processors.

4.This solution is a more accessible way of receiving good results as it can be activated at a higher level, namely at the level of configuring the switch software and replicating Short on standard machines. Further improvements could be made if higher performance equipment was used. Cost is generally an important concern. The design proposed in this research benefits the network security requirements at low cost.

**4.1.5 Modules Description:**

**QoS Classification And Policy Methods:**

Classification is the process of identifying the data packets to a class or group in order to manage the packet appropriately. QoS features such as a policy map and class map can be used to achieve this. The class information can be assigned by switch, router, or end host. Policing involves creating a policy that defines a group weight (the number of bytes to be processed together) for the traffic and applies it to the interface. Policing can be applied to a packet per direction and can occur on the ingress and egress interfaces. Different types of traffic can be recognized in terms of type, and ports and differentiated policies can be set accordingly

**Parallel Technology with QoS:**

Parallel NIDPS is a form of computation where many NIDPS nodes work simultaneously, operating on the principle that the large incoming data can be divided into smaller sets, which are processed at the same time. Parallelism of NIDPS can occur at three general levels: the high-level processing node (entire system), the component level (specific tasks are isolated and parallelized) and the sub-component level parallelism (function within a specific task)

**QoS Classification, Policing And Marking For Ingress And Egress Interfaces (Queues):** Queues, class, and policy technologies can use access control lists (ACLs) to allow the processing management of different types and patterns of incoming and outgoing packets. The novel configuration proposed in this paper uses an ACL technology with a class map and SVI

Queues, as well as a policy map that specifies each type of IP traffic (e.g., ICMP, TCP and UDP) to be processed by implementing parallel output queues with associated parallel NIDPS nodes

**4.2 Feasibility Study**

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* Technical Feasibility
* Operational Feasibility
* Economical Feasibility

**4.2.1 Economic Feasibility**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

**4.2.2 Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

* Is there sufficient support for the management from the users?
* Will the system be used and work properly if it is being developed and implemented?
* Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

**4.2.3 Technical Feasibility**

The technical issue usually raised during the feasibility stage of the investigation includes the following:

* Does the necessary technology exist to do what is suggested?
* Do the proposed equipments have the technical capacity to hold the data required to use the new system?
* Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
* Can the system be upgraded if developed?
* Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source.

**5. System Requirements Specification:**

**5.1 Introduction**

A **Software Requirements Specification** (**SRS**) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) – is a complete description of the behavior of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

**System requirements specification:** A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development life cycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [**Business requirements**](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* **Product requirements** describe properties of a system or product (which could be one of

several ways to accomplish a set of business requirements.)

* **Process requirements** describe activities performed by the developing organization. For instance, process requirements could specify specific methodologies that must be followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, a maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement); a requirement that the product be maintainable (a Product requirement) often is addressed by imposing requirements to follow particular development styles

**5.2 Purpose**

An systems engineering, a **requirement** can be a description of what a system must do, referred to as a [Functional Requirement](http://en.wikipedia.org/wiki/Functional_requirements). This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements), or 'performance requirements' or 'quality of service requirements.' Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability.

A collection of requirements define the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying how the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality (e.g. of user interfaces) with other owned products.

In software engineering, the same meanings of requirements apply, except that the focus of interest is the software itself.

**5.3 FUNCTIONAL Requirements**

**1. Data**

**2. Classiication**

**3. Policy**

**4. Ingress**

**5. Egress**

**5.4 NON Functional Requirements**

The major non-functional Requirements of the system are as follows

**Usability**

The system is designed with completely automated process hence there is no or less user intervention.

**Reliability**

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

**Performance**

This system is developing in the high level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

**Supportability**

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having JVM, built into the system.

**Implementation**

The system is implemented in web environment using struts framework. The apache tomcat is used as the web server and windows xp professional is used as the platform.

Interface the user interface is based on Struts provides HTML Tag.

**5.5 Input & Output Design**

**Input Design:**

Inaccurate input data are the most common causes of errors in data processing. Errors entered by data entry operators can be controlled by the Input design. "Input design is the process of converting user originated inputs to computer based formats". It consists of developing specification and procedure for data preparation.

**Objectives of input design:**

The main objectives of input design are:

1. Controlling amount of input: Due to so many reasons, design should control the quantity of data    for input. Reducing the data requirement can lower cost by reducing labour expenses. By reducing input requirement, the analyst can speed the entire process from data capture to providing results to the users.

2. Avoiding delay: A processing delay resulting from data preparation or data entry operator is called bottleneck. Avoiding bottleneck should always be one objective of the analyst while designing output.

3. Avoiding errors in data: The rate at which errors occurs depends on the quantity of data, i.e. smaller the amount of data to input the fewer the opportunities for errors.

4. Keeping the process simple: Simplicity works and is accepted by the users. Complexity should be avoided when there are simple alternatives.

**Output Design:**

The term output necessarily implies to information on printed or displayed by an information system. Following are the activities that are carried out in output design stage.

Ø  Identification of specific output required to meet the information requirements.

Ø  Selecting of methods for processing outputs.

Ø  Designing of reports, formats or other documents that acts as a carrier of information.

**Output Design Activities**

The output design of an information system must meet the following objectives:

1. The output design should provide information about the past, present or future events. The operational control level outputs provide operations of the past and present events. On the other hand, strategic planning level provides information of the future events.

2. The output design should indicate the important events, opportunities and problems.

3. The output design should be designed keeping in mind that an action must be triggered in response to some event. A set of rule is pre- designed for such trigger.

4. The output design should produce some action to the transaction. For e.g. when the telephone bill is generated, a receipt is printed.

**5.6 Hardware Requirements:**

Processor : Pentium IV

Hard Disk : 500GB

RAM : 2GB or more

**5.7 Software Requirements:**

Operating System : Windows XP/2003 or Linux (Any OS)

User Interface : HTML, CSS

Client-side Scripting : JavaScript

Programming Language : Java

Web Applications : JDBC, Servlets, JSP

IDE/Workbench : My Eclipse 8.6

Database : Oracle 11g

Server Deployment : Tomcat 7.0

**6. System Design:**

**6.1 Introduction**

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed, design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affection the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design.

System Design also called top-level design aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of the system design all the major data structures, file formats, output formats, and the major modules in the system and their specifications are decided.

During, Detailed Design, the internal logic of each of the modules specified in system design is decided. During this phase, the details of the data of a module is usually specified in a high-level design description language, which is independent of the target language in which the software will eventually be implemented.

In system design the focus is on identifying the modules, where as during detailed design the focus is on designing the logic for each of the modules. In other works, in system design the attention is on what components are needed, while in detailed design how the components can be implemented in software is the issue.

Design is concerned with identifying software components specifying relationships among components. Specifying software structure and providing blue print for the document phase. Modularity is one of the desirable properties of large systems. It implies that the system is divided into several parts. In such a manner , the interaction between parts is minimal clearly specified.

During the system design activities , Developers bridge the gap between the requirements specification , produced during requirements elicitation and analysis , and the system that is delivered to the user.

Design is the place where the quality is fostered in development . Software design is a process through which requirements are translated into a representation of software.

**6.2 System Model**

**Introduction to UML**

The unified Modeling Language (UML) is a standard language for writing software blueprints. The UML may be used to visualize, specify , construct and document the artifacts of software-intensive system.

The goal of UML is to provide a standard notation that can be used by all object - oriented methods and to select and integrate the best elements .UML is itself does not prescribe or advice on how to use that notation in a software development process or as part of an object - design methodology. The UML is more than just bunch of graphical symbols. Rather , behind each symbol in the UML notation is well-defined semantics.

The system development focuses on three different models of the system.

* Functional model
* Object model
* Dynamic model

**Functional model** in UML is represented with use case diagrams , describing the functionality of the system from user point of view.

**Object model** in UML is represented with class diagrams , describing the structure of the system in terms of objects , attributes , associations and operations.

**Dynamic model**  in UML is represented with sequence diagrams , start chart diagrams and activity diagrams describing the internal behaviour of the system.

**Scenarios**

A Use Case is an abstraction that all describes all possible scenarios involving the described functionality . A scenario is an instance of a use case describing a concrete set of actions.

* The **name** of the scenario enables us to refer it ambiguously. The name of scenario is underlined to indicate it is an instance.
* The **Participating actor instance** field indicates which actor instance are involved in this scenario. Actor instance also have underlined names.

The **Flow of Events** of scenario describe the sequence of events step by step.

**6.3 Data Flow Diagram:**

A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes, may be described logically and independently of the physical components associated with the system. The DFD is also know as a data flow graph or a bubble chart.

DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during design activity this is taken as the basis for drawing the system’s structure charts. The Basic Notation used to create a DFD’s are as follows:

**1. Dataflow:** Data move in a specific direction from an origin to a destination.

**2. Process:** People, procedures, or devices that use or produce (Transform) Data. The physical component is not identified.

**3. Source:** External sources or destination of data, which may be People, programs, organizations or other entities.

**4. Data Store:** Here data are stored or referenced by a process in the System.

**CONTEXT LEVEL 0 DIAGRAM:**

**CONTEXT LEVEL DATAFLOW DIAGRAM**

**DATABASE**

**Data O/p stage**

**Switch**

**UI Screens**

**Data i/p stage Data O/p stage**

**Reports**

**Data o/p stage**

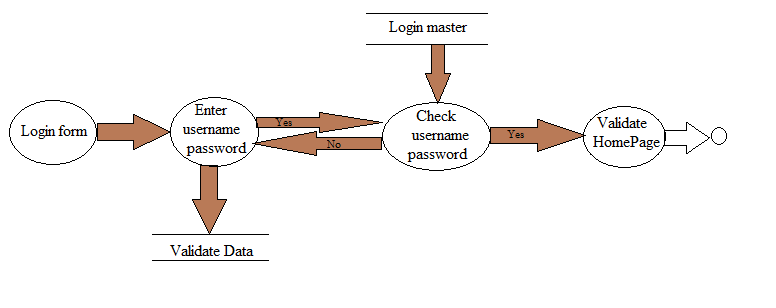
**A New Architecture for Network Intrusion Detection and Prevention**

a

**System Process**

**Context level1 Diagram:**

**Login DFD**



**Context level 2:**

**Switch**

**UI Displays set of operations**

**6.4 UML Diagrams:**

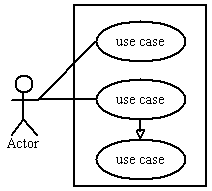
**What is a UML Use Case Diagram?**

Use case diagrams model the functionality of a system using actors and use cases. Use cases are services or functions provided by the system to its users.

**Basic Use Case Diagram Symbols and Notations**

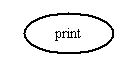
**System**

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



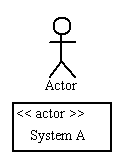
**Use Case**

Draw use cases using ovals. Label with ovals with verbs that represent the system's functions.



**Actors**

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



**Relationships**

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.

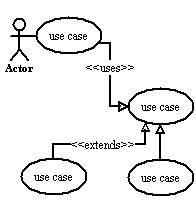




Fig: Use Case diagram

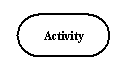
**Activity Diagram**

An activity diagram illustrates the dynamic nature of a system by modeling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation. Because an activity diagram is a special kind of state chart diagram, it uses some of the same modeling conventions.

**Basic Activity Diagram Symbols and Notations**

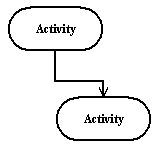
**Action states**

Action states represent the non interruptible actions of objects. You can draw an action state in Smart Draw using a rectangle with rounded corners.



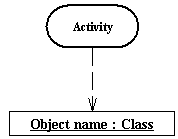
**Action Flow**

Action flow arrows illustrate the relationships among action states.



**Object Flow**

Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object.



**Initial State**

A filled circle followed by an arrow represents the initial action state.

Initial State

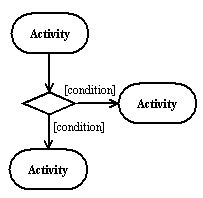
**Final State**

An arrow pointing to a filled circle nested inside another circle represents the final action state.

Final State

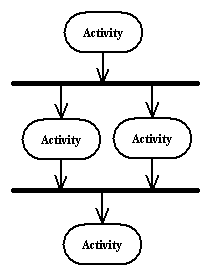
**Branching**

A diamond represents a decision with alternate paths. The outgoing alternates should be labeled with a condition or guard expression. You can also label one of the paths "else."



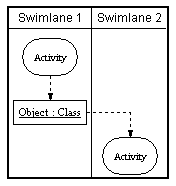
**Synchronization**

A synchronization bar helps illustrate parallel transitions. Synchronization is also called forking and joining.



**Swim lanes**

Swim lanes group related activities into one column.



****

Fig: Activity diagram

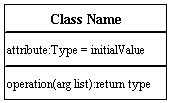
**Class Diagram**

Class diagrams are the backbone of almost every object-oriented method including UML. They describe the static structure of a system.

**Basic Class Diagram Symbols and Notations**

Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.

Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition, and  
Write operations into the third.



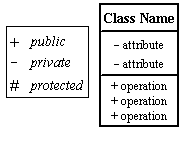
**Active Class**

Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.



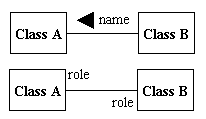
**Visibility**

Use visibility markers to signify who can access the information contained within a class. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class.



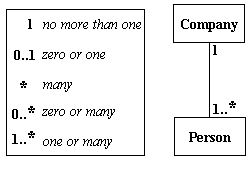
**Associations**

Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes see each other.  
***Note:*** It's uncommon to name both the association and the class roles.



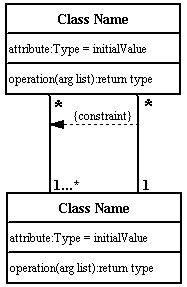
**Multiplicity (Cardinality)**

Place multiplicity notations near the ends of an association. These symbols indicate the number of instances of one class linked to one instance of the other class. For example, one company will have one or more employees, but each employee works for one company only.



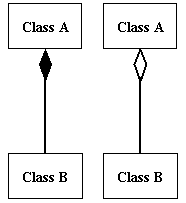
**Constraint**

Place constraints inside curly braces {}.

http://wc1.smartdraw.com/resources/tutorials/images/uml_constraint.gif*Simple Constraint* 

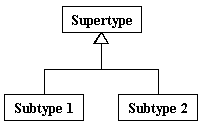
**Composition and Aggregation**

Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole, and Class B, its part. Illustrate **composition** with a filled diamond. Use a hollow diamond to represent a simple **aggregation** relationship, in which the "whole" class plays a more important role than the "part" class, but the two classes are not dependent on each other. The diamond end in both a composition and aggregation relationship points toward the "whole" class or the aggregate



**Generalization**

Generalization is another name for inheritance or an "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. For example, Honda is a type of car. So the class Honda would have a generalization relationship with the class car.



In real life coding examples, the difference between inheritance and aggregation can be confusing. If you have an aggregation relationship, the aggregate (the whole) can access only the PUBLIC functions of the part class. On the other hand, inheritance allows the inheriting class to access both the PUBLIC and PROTECTED functions of the super class.



Fig: class Diagram.

**Sequence Diagram**

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time.

**Basic Sequence Diagram Symbols and Notations**

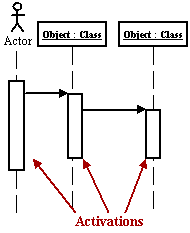
**Class roles**

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.

Class roles

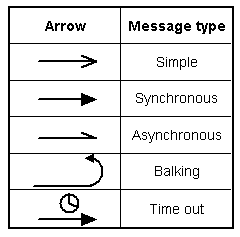
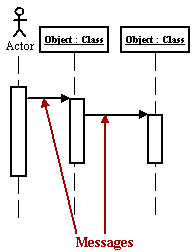
**Activation**

Activation boxes represent the time an object needs to complete a task.



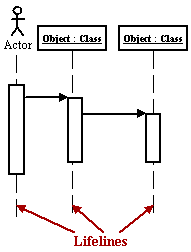
**Messages**

Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

  
*Various message types for Sequence and Collaboration diagrams*

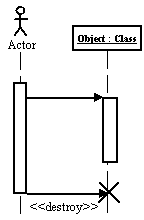
**Lifelines**

Lifelines are vertical dashed lines that indicate the object's presence over time.



**Destroying Objects**

Objects can be terminated early using an arrow labeled "<< destroy >>" that points to an X.



**Loops**

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets.

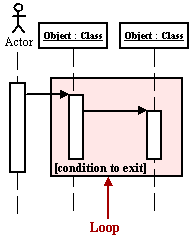




Fig: Sequence Diagram

**Collaboration Diagram**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.

**Basic Collaboration Diagram Symbols and Notations**

**Class roles**

Class roles describe how objects behave. Use the UML object symbol to illustrate class roles, but don't list object attributes.

Class roles

**Association roles**

Association roles describe how an association will behave given a particular situation. You can draw association roles using simple lines labeled with stereotypes.

Association roles

**Messages**

Unlike sequence diagrams, collaboration diagrams do not have an explicit way to denote time and instead number messages in order of execution. Sequence numbering can become nested using the Dewey decimal system. For example, nested messages under the first message are labeled 1.1, 1.2, 1.3, and so on. The a condition for a message is usually placed in square brackets immediately following the sequence number. Use a \* after the sequence number to indicate a loop.

[Learn how to add arrows to your lines.](http://www.smartdraw.com/resources/tutorials/Lines)

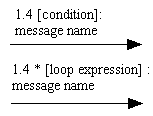




Fig: Collaboration Diagram

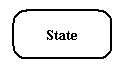
**State chart Diagram**

A state chart diagram shows the behavior of classes in response to external stimuli. This diagram models the dynamic flow of control from state to state within a system.

**Basic State chart Diagram Symbols and Notations**

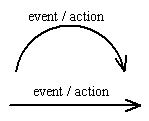
**States**

States represent situations during the life of an object. You can easily illustrate a state in Smart Draw by using a rectangle with rounded corners.



**Transition**

A solid arrow represents the path between different states of an object. Label the transition with the event that triggered it and the action that results from it.



**Initial State**

A filled circle followed by an arrow represents the object's initial state.

Initial State

**Final State**

An arrow pointing to a filled circle nested inside another circle represents the object's final state.

Final State

**Synchronization and Splitting of Control**

A short heavy bar with two transitions entering it represents a synchronization of control. A short heavy bar with two transitions leaving it represents a splitting of control that creates multiple states.

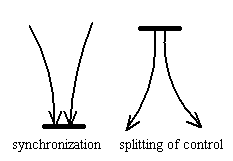




Fig: State Chart diagram

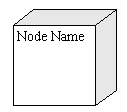
**Deployment Diagram**

Deployment diagrams depict the physical resources in a system including nodes, components, and connections.

**Basic Deployment Diagram Symbols and Notations**

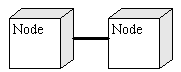
**Component**

A node is a physical resource that executes code components.  
[Learn how to resize grouped objects like nodes.](http://www.smartdraw.com/resources/tutorials/Objects)



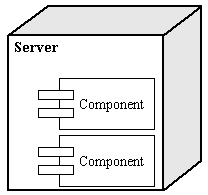
**Association**

Association refers to a physical connection between nodes, such as Ethernet.  
[Learn how to connect two nodes.](http://www.smartdraw.com/resources/tutorials/Lines)



**Components and Nodes**

Place components inside the node that deploys them.



**DEPLOYMENT DIAGRAM:**



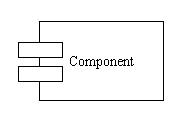
**Component Diagram**

A component diagram describes the organization of the physical components in a system.

**Basic Component Diagram Symbols and Notations**

**Component**

A component is a physical building block of the system. It is represented as a rectangle with tabs.  
[Learn how to resize grouped objects like components.](http://www.smartdraw.com/resources/tutorials/Objects)



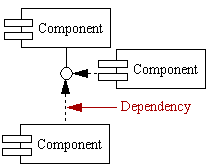
**Interface**

An interface describes a group of operations used or created by components.

Interface

**Dependencies**

Draw dependencies among components using dashed arrows.  
[Learn about line styles in SmartDraw.](http://www.smartdraw.com/resources/tutorials/Lines)



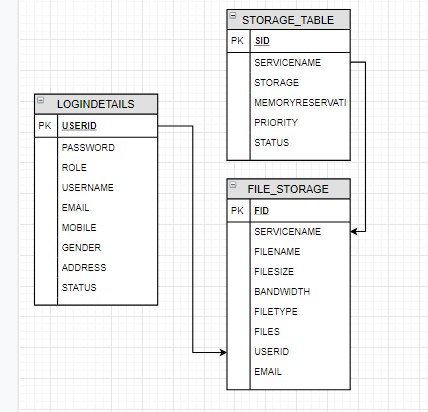
**COMPONENT DIAGRAM:**

****

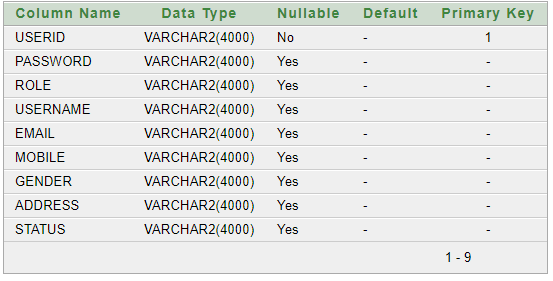
**6.5 Data Dictionaries and ER Diagram**

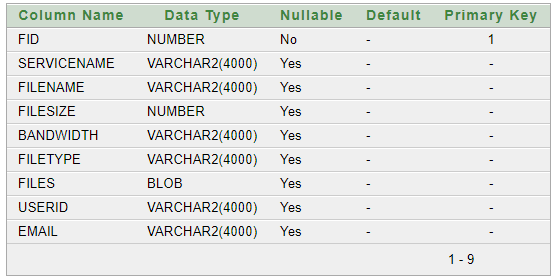
E-R DIAGRAM:

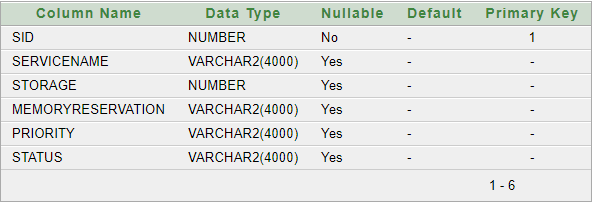
ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.



**Data dictionary:**

****

****

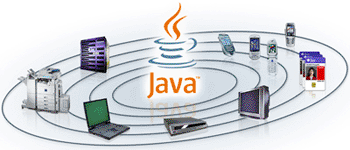
****

**7. Implementation:**

**7.1 Technology Description:**

**About the Java Technology**

The Java platform consists of the Java application programming interfaces (APIs) and the Java virtual machine (JVM).



The following Java technology lets developers, designers, and business partners develop and deliver a consistent user experience, with one environment for applications on mobile and embedded devices. Java meshes the power of a rich stack with the ability to deliver customized experiences across such devices.

Java APIs are libraries of compiled code that you can use in your programs. They let you add ready-made and customizable functionality to save you programming time.  
Java programs are run (or interpreted) by another program called the Java Virtual Machine. Rather than running directly on the native operating system, the program is interpreted by the Java VM for the native operating system. This means that any computer system with the Java VM installed can run Java programs regardless of the computer system on which the applications were originally developed.

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris TM Operating System (Solaris OS), Linux, or Mac OS.

Java technology is both a programming language and a platform.

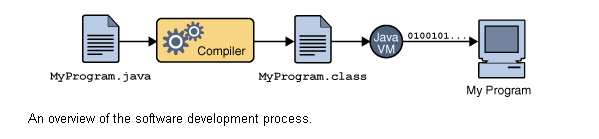
**The Java Programming Language**

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

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

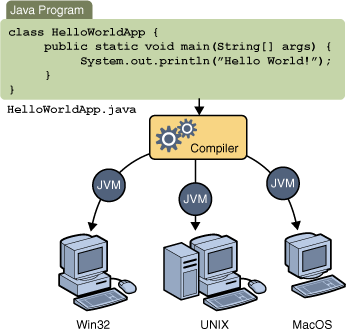
Each of the preceding buzzwords is explained in [The Java Language Environment](http://java.sun.com/docs/white/langenv/) , a white paper written by James Gosling and Henry McGilton.

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine[1](http://download.oracle.com/javase/tutorial/getStarted/intro/definition.html#FOOT) (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.



An overview of the software development process.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris™ Operating System (Solaris OS), Linux, or Mac OS. Some virtual machines, such as the [Java HotSpot virtual machine](http://java.sun.com/products/hotspot/), perform additional steps at runtime to give your application a performance boost. This include various tasks such as finding performance bottlenecks and recompiling (to native code) frequently used sections of code



Through the Java VM, the same application is capable of running on multiple platforms.

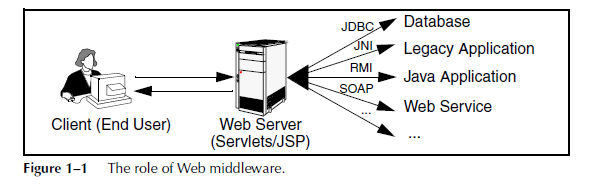
**Servlet and JSP technology**

Servlet and JSP technology has become the technology of choice for developing online stores, interactive

**A Servlet’s Job**

Servlets are Java programs that run on Web or application servers, acting as a middle layer between requests coming from Web browsers or other HTTP clients and databases or applications on the HTTP server. Their job is to perform the following tasks,

as illustrated in Figure 1–1.



1. **Read the explicit data sent by the client.**

The end user normally enters this data in an HTML form on a Web page. However, the data could also come from an applet or a custom HTTP client program. Chapter 4 discusses how servlets read this data.

2. **Read the implicit HTTP request data sent by the browser.**

Figure 1–1 shows a single arrow going from the client to the Web server (the layer where servlets and JSP execute), but there are really two varieties of data: the explicit data that the end user enters in a form and the behind-the-scenes HTTP information. Both varieties are critical. The HTTP information includes cookies, information about media types and compression schemes the browser understands,

3. **Generate the results.**

This process may require talking to a database, executing an RMI or EJB call, invoking a Web service, or computing the response directly. Your real data may be in a relational database. Fine. But your database probably doesn’t speak HTTP or return results in HTML, so the Web browser can’t talk directly to the database. Even if it could, for security reasons, you probably would not want it to. The same argument applies to most other applications. You need the Web middle layer to extract the incoming data from the HTTP stream, talk to the application, and embed the results inside a document.

4. **Send the explicit data (i.e., the document) to the client.**

This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), or even a compressed format like gzip that is layered on top of some other underlying format. But, HTML is by far the most common format, so an important servlet/JSP task is to wrap the results inside of HTML.

5. **Send the implicit HTTP response data.**

Figure 1–1 shows a single arrow going from the Web middle layer (the servlet or JSP page) to the client. But, there are really two varieties of data sent: the document itself and the behind-the-scenes HTTP information. Again, both varieties are critical to effective development. Sending HTTP response data involves telling the browser or other client what type of document is being returned (e.g., HTML), setting cookies and caching parameters

**The Advantages of Servlets Over “Traditional” CGI**

Java servlets are more efficient, easier to use, more powerful, more portable, safer, and cheaper than traditional CGI and many alternative CGI-like technologies. With traditional CGI, a new process is started for each HTTP request. If the CGI program itself is relatively short, the overhead of starting the process can dominate the execution time. With servlets, the Java virtual machine stays running and handles each request with a lightweight Java thread, not a heavyweight operating system process. Similarly, in traditional CGI, if there are N requests to the same CGI program, the code for the CGI program is loaded into memory N times. With servlets, however, there would be N threads, but only a single copy of the servlet class would be

loaded. This approach reduces server memory requirements and saves time by instantiating fewer objects. Finally, when a CGI program finishes handling a request, the program terminates. This approach makes it difficult to cache computations, keep database connections open, and perform other optimizations that rely on persistent data. Servlets, however, remain in memory even after they complete a response, so it is straightforward to store arbitrarily complex data between client requests.

**Convenient**

Servlets have an extensive infrastructure for automatically parsing and decoding HTML form data, reading and setting HTTP headers, handling cookies, tracking sessions, and many other such high-level utilities. In CGI, you have to do much of this yourself. Besides, if you already know the Java programming language, why learn Perl too? You’re already convinced that Java technology makes for more reliable and reusable code than does Visual Basic, VBScript, or C++. Why go back to those languages for server-side programming?

**Powerful**

Servlets support several capabilities that are difficult or impossible to accomplish with regular CGI. Servlets can talk directly to the Web server, whereas regular CGI programs cannot, at least not without using a server-specific API. Communicating with the Web server makes it easier to translate relative URLs into concrete path names, for instance. Multiple servlets can also share data, making it easy to implement database connection pooling and similar resource-sharing optimizations. Servlets can also maintain information from request to request, simplifying techniques like session tracking and caching of previous computations.

**Portable**

Servlets are written in the Java programming language and follow a standard API. Servlets are supported directly or by a plug-in on virtually every major Web server. Consequently, servlets written for, say, Macromedia Run can run virtually unchanged on Apache Tomcat, Microsoft Internet Information Server (with a separate plug-in), IBM Web Sphere, planet Enterprise Server, Oracle9i AS, or Star Nine Webster. They are part of the Java 2 Platform, Enterprise Edition (J2EE; see <http://java.sun.com/j2ee/>), so industry support for servlets is becoming even more pervasive.

**Inexpensive**

A number of free or very inexpensive Web servers are good for development use or deployment of low- or medium-volume Web sites. Thus, with servlets and JSP you can start with a free or inexpensive server and migrate to more expensive servers with high-performance capabilities or advanced administration utilities only after your project meets initial success. This is in contrast to many of the other CGI alternatives, which require a significant initial investment for the purchase of a proprietary package. Price and portability are somewhat connected. For example, Marty tries to keep track of the countries of readers that send him questions by email. India was near the top of the list, probably #2 behind the U.S. Marty also taught one of his JSP and servlet

training courses (see http://courses.coreservlets.com/) in Manila, and there was great interest in servlet and JSP technology there. Now, why are India and the Philippines both so interested? We surmise that the answer is twofold. First, both countries have large pools of well-educated software developers.

Second both countries have (or had, at that time) highly unfavorable currency exchange rates against the U.S. dollar. So, buying a special-purpose Web server from a U.S. company consumed a large part of early project funds. But, with servlets and JSP, they could start with a free server: Apache Tomcat (either standalone, embedded in the regular Apache Web server, or embedded in Microsoft IIS). Once the project starts to become successful, they could move to a

server like Caucho Resin that had higher performance and easier administration but that is not free. But none of their servlets or JSP pages have to be rewritten. If their project becomes even larger, they might want to move to a distributed (clustered) environment. No problem: they could move to Macromedia Run Professional, which supports distributed applications (Web farms). Again, none of their servlets or JSP pages have to be rewritten. If the project becomes quite large and complex, they might want to use Enterprise JavaBeans (EJB) to encapsulate their business logic. So, they might switch to BEA Web Logic or Oracle9i AS. Again, none of their servlets

or JSP pages have to be rewritten. Finally, if their project becomes even bigger, they might move it off of their Linux box and onto an IBM mainframe running IBM Web- Sphere. But once again, none of their servlets or JSP pages have to be rewritten

**Secure**

One of the main sources of vulnerabilities in traditional CGI stems from the fact that the programs are often executed by general-purpose operating system shells. So, the CGI programmer must be careful to filter out characters such as backquotes and semicolons that are treated specially by the shell. Implementing this precaution is harder than one might think, and weaknesses stemming from this problem are constantly being uncovered in widely used CGI libraries. A second source of problems is the fact that some CGI programs are processed by languages that do not automatically check array or string bounds. For example, in C and C++ it is perfectly legal to allocate a 100-element array and then write into the 999th “element,” which is really some random part of program memory. So, programmers who forget to perform this check open up their system to deliberate or accidental buffer overflow attacks. Servlets suffer from neither of these problems. Even if a servlet executes a system call (e.g., with Runtime. Exec or JNI) to invoke a program on the local operating system, it does not use a shell to do so. And, of course, array bounds checking and other memory protection features are a central part of the Java programming language.

**Mainstream**

There are a lot of good technologies out there. But if vendors don’t support them and developers don’t know how to use them, what good are they? Servlet and JSP technology is supported by servers from Apache, Oracle, IBM, Sybase, BEA, Macromedia, Caucho, Sun/planet, New Atlanta, ATG, Fujitsu, Ultras, Silver stream, the World Wide Web Consortium (W3C), and many others. Several low-cost plugins add support to Microsoft IIS and Zeus as well. They run on Windows, Unix/Linux, Maces, VMS, and IBM mainframe operating systems. They are the single most popular application of the Java programming language. They are arguably the most popular choice for developing medium to large Web applications. They are used by the airline

industry (most United Airlines and Delta Airlines Web sites), e-commerce (ofoto.com), online banking (First USA Bank, Blanco Popular de Puerto Rico), Web search engines/portals (excite.com), large financial sites (American Century Investments), and hundreds of other sites that you visit every day. Of course, popularity alone is no proof of good technology. Numerous

counter-examples abound. But our point is that you are not experimenting with a

new and unproven technology when you work with server-side Java.

**The Role of JSP**

A somewhat oversimplified view of servlets is that they are Java programs with HTML embedded inside of them. A somewhat oversimplified view of JSP documents is that they are HTML pages with Java code embedded inside of them. For example, compare the sample servlet shown earlier (Listing 1.1) with the JSP page shown below (Listing 1.2). They look totally different; the first looks mostly like a regular Java class, whereas the second looks mostly like a normal HTML page. The interesting thing is that, despite the huge apparent difference, behind the scenes they are the same. In fact, a JSP document is just another way of writing a servlet. JSP pages get translated into servlets, the servlets get compiled, and it is the servlets that run at request time. So, the question is, If JSP technology and servlet technology are essentially equivalent in power, does it matter which you use? The answer is, Yes, yes, yes! The issue is not power, but convenience, ease of use, and maintainability. For example, anything you can do in the Java programming language you could do in assembly language. Does this mean that it does not matter which you use? Hardly. JSP is discussed in great detail starting in Chapter 10. But, it is worthwhile mentioning now how servlets and JSP fit together. JSP is focused on simplifying the creation and maintenance of the HTML. Servlets are best at invoking the business logic and performing complicated operations. A quick rule of thumb is that servlets are best for tasks oriented toward processing, whereas JSP is best for tasks oriented toward presentation. For some requests, servlets are the right choice. For other requests, JSP is a better option. For still others, neither servlets alone nor JSP alone is best, and a combination of the two (see Chapter 15, “Integrating Servlets and JSP: The Model View Controller (MVC) Architecture”) is best. But the point is that you need both servlets and JSP in your overall project: almost no project will consist

entirely of servlets or entirely of JSP. You want both.

**8. Coding:**

**Registration.jsp**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"utf-8"*>

<title>Form-v8 by Colorlib</title>

<!-- Mobile Specific Metas -->

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1, maximum-scale=1"*>

<!-- Font-->

<link rel=*"stylesheet"* type=*"text/css"* href=*"css/sourcesanspro-font.css"*>

<!-- Main Style Css -->

<link rel=*"stylesheet"* href=*"css/style2.css"*/>

</head>

<body class=*"form-v8"*>

<jsp:include page=*"HomeMenu.jsp"*></jsp:include>

<div class=*"page-content"*>

<div class=*"form-v8-content"*>

<!--<div class="form-left">

<img src="images/form-v8.jpg" alt="form">

</div>

--><div class=*"form-right"*>

<div class=*"tab"*>

<div class=*"tab-inner"*>

<button class=*"tablinks"* onclick="openCity(event, 'sign-up')" id=*"defaultOpen"*>Sign Up</button>

</div>

</div>

<form class=*"form-detail"* action=*"Registerservlet"* >

<center><label><font color=*"red"* size=*"4"*><% String status=request.getParameter("status"); **if**(status!=**null**){out.println(status);} %></font></label> </center>

<div class=*"tabcontent"* id=*"sign-up"*>

<div class=*"form-row"*>

<label class=*"form-row-inner"*>

<input type=*"text"* name=*"uid"* id=*"full\_name"* class=*"input-text"* required>

<span class=*"label"*>Userid</span>

<span class=*"border"*></span>

</label>

</div>

<div class=*"form-row"*>

<label class=*"form-row-inner"*>

<input type=*"password"* name=*"pwd"* id=*"password"* class=*"input-text"* required>

<span class=*"label"*>Password</span>

<span class=*"border"*></span>

</label>

</div>

<div class=*"form-row"*>

<label class=*"form-row-inner"*>

<input type=*"text"* name=*"name"* id=*"full\_name"* class=*"input-text"* required>

<span class=*"label"*>Username</span>

<span class=*"border"*></span>

</label>

</div>

<div class=*"form-row"*>

<label class=*"form-row-inner"*>

<input type=*"text"* name=*"email"* id=*"your\_email"* class=*"input-text"* required>

<span class=*"label"*>E-Mail</span>

<span class=*"border"*></span>

</label>

</div>

<div class=*"form-row"*>

<select name=*"g"* class=*"input-text"* required>

<option value=*""*>

Please select Gender

</option>

<option value=*"male"*>

Male

</option>

<option value=*"Female"*>

Female

</option>

</select>

</div>

<div class=*"form-row"*>

<label class=*"form-row-inner"*>

<input type=*"text"* name=*"mob"* id=*"your\_email"* class=*"input-text"* required>

<span class=*"label"*>Mobile</span>

<span class=*"border"*></span>

</label>

</div>

<div class=*"form-row"*>

<label class=*"form-row-inner"*>

<input type=*"text"* name=*"add"* id=*"your\_email"* class=*"input-text"* required>

<span class=*"label"*>Address</span>

<span class=*"border"*></span>

</label>

</div>

<div class=*"form-row-last"*>

<input type=*"submit"* name=*"register"* class=*"register"* value=*"Register"*>

</div>

</div>

</form>

</div>

</div>

</div>

<script type=*"text/javascript"*>

**function** **openCity**(evt, cityName) {

**var** i, tabcontent, tablinks;

tabcontent = document.getElementsByClassName("tabcontent");

**for** (i = 0; i < tabcontent.length; i++) {

tabcontent[i].style.display = "none";

}

tablinks = document.getElementsByClassName("tablinks");

**for** (i = 0; i < tablinks.length; i++) {

tablinks[i].className = tablinks[i].className.replace(" active", "");

}

document.getElementById(cityName).style.display = "block";

evt.currentTarget.className += " active";

}

// Get the element with id="defaultOpen" and click on it

document.getElementById("defaultOpen").click();

</script>

</body><!-- This templates was made by Colorlib (https://colorlib.com) -->

</html>

**Registrationservlet:**

package com.archi.action;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.archi.dao.RegistrationDao;

import com.archi.dto.Profilebean;

public class Registerservlet extends HttpServlet {

/\*\*

\* The doPost method of the servlet. <br>

\*

\* This method is called when a form has its tag value method equals to post.

\*

\* @param request the request send by the client to the server

\* @param response the response send by the server to the client

\* @throws ServletException if an error occurred

\* @throws IOException if an error occurred

\*/

private final String UPLOAD\_DIRECTORY = "C:/uploads";

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

Profilebean pb = new Profilebean();

try {

pb.setUserid(request.getParameter("uid"));

pb.setPassword(request.getParameter("pwd"));

pb.setUsername(request.getParameter("name"));

pb.setEmail(request.getParameter("email"));

pb.setGender(request.getParameter("g"));

pb.setMobile(request.getParameter("mob"));

pb.setAddress(request.getParameter("add"));

RegistrationDao rdo=new RegistrationDao();

int i=rdo.register(pb);

if(i!=0){

RequestDispatcher rd=request.getRequestDispatcher("Login.jsp?status=Registration Successfully Completed ");

rd.include(request, response);

}

else {

RequestDispatcher rd=request.getRequestDispatcher("Registration.jsp?status=Registrtion Fail Try Again ");

rd.include(request, response);

}

}

catch (Exception e) {

e.printStackTrace();

RequestDispatcher rd=request.getRequestDispatcher("Registration.jsp?status=Some Internal Problem");

rd.include(request, response);

}

}

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doPost(request, response);

}

}

**RegistrationDao:**

package com.archi.dao;

import java.io.FileInputStream;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import com.archi.cryptoutill.CryptMsgUtil;

import com.archi.db.DbCon;

import com.archi.dto.Profilebean;

public class RegistrationDao extends DbCon {

public int register(Profilebean pb) {

int i=0;

String userid=" ";

Connection con=null;

con=getConnection();

System.out.println("connection post\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"+con);

try {

PreparedStatement ps1=con.prepareStatement("select USERID from LOGINDETAILS");

ResultSet rs=ps1.executeQuery();

while(rs.next()){

userid=rs.getString(1);

}

if(userid.equals(pb.getUserid())){

return i=0;

}

else {

PreparedStatement pstmt=con.prepareStatement("insert into LOGINDETAILS(USERID,PASSWORD,ROLE,USERNAME,EMAIL,MOBILE,ADDRESS,GENDER,STATUS) values(?,?,?,?,?,?,?,?,?) ");

pstmt.setString(1, pb.getUserid());

pstmt.setString(2, pb.getPassword());

pstmt.setString(3, "User");

pstmt.setString(4, pb.getUsername());

pstmt.setString(5, pb.getEmail());

pstmt.setString(6, pb.getMobile());

pstmt.setString(7, pb.getAddress());

pstmt.setString(8, pb.getGender());

pstmt.setString(9, "Active");

i=pstmt.executeUpdate();

System.out.println(i+"in i Record is Inserted successfully");

con.close();

}

} catch (Exception e) {

e.printStackTrace();

}

return i;

}

public ArrayList<Profilebean> login(Profilebean pb) {

Connection con=null;

//int i=0;

Profilebean pb1=null;

con=getConnection();

ArrayList<Profilebean> list=new ArrayList<Profilebean>();

try {

PreparedStatement ps=con.prepareStatement("select ROLE,USERNAME,EMAIL from LOGINDETAILS where USERID=? and PASSWORD=? ");

ps.setString(1, pb.getUserid());

ps.setString(2, pb.getPassword());

ResultSet rs=ps.executeQuery();

while(rs.next()){

pb1=new Profilebean();

pb1.setRole(rs.getString(1));

pb1.setUsername(rs.getString(2));

pb1.setEmail(rs.getString(3));

list.add(pb1);

}

con.close();

} catch (SQLException e) {

e.printStackTrace();

}

return list;

}

**9. System Testing:**

**9.1 Testing Methodologies**

Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior implemented system. From modeling point of view , testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems.

The process of executing a program with intent of finding errors is called testing. During testing , the program to be tested is executed with a set of test cases , and the output of the program for the test cases is evaluated to determine if the program is performing as expected . Testing forms the first step in determining the errors in the program. The success of testing in revealing errors in program depends critically on test cases.

**Strategic Approach to Software Testing:**

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirements analysis where the information domain , functions , behavior , performance , constraints and validation criteria for software are established. moving inward along the spiral , we come to design and finally to coding . To develop computer software we spiral in along streamlines that decreases the level of abstraction on each item.

A Strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing will progress by moving outward along the spiral to integration testing , where the focus on the design and the concentration of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed . Finally we arrive at system testing , where the software and other system elements are tested as a whole .

UNUNI

UNIT TESTING

MODULE

SUB-SYSTEM

**Component**

SYSTEM TESTING

**Integration Testing**

ACCEPTANCE

**User Testing**

**Different Levels of Testing**

Client Needs Acceptance Testing

Requirements System Testing

Design Integration Testing

Code Unit Testing

Testing is the process of finding difference between the expected behavior specified by system models and the observed behavior of the implemented system.

**Testing Activities**

Different levels of testing are used in the testing process , each level of testing aims to test different aspects of the system. the basic levels are:

Unit testing

Integration testing

System testing

Acceptance testing

**Unit Testing**

Unit testing focuses on the building blocks of the software system, that is, objects and sub system. There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall tests activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults given that few components are involved in this test. Third, Unit testing allows parallelism in the testing activities, that is each component can be tested independently of one another. Hence the goal is to test the internal logic of the module.

**Integration Testing**

In the integration testing, many test modules are combined into sub systems , which are then tested . The goal here is to see if the modules can be integrated properly, the emphasis being on testing module interaction.

After structural testing and functional testing we get error free modules. These modules are to be integrated to get the required results of the system. After checking a module, another module is tested and is integrated with the previous module. After the integration, the test cases are generated and the results are tested.

**System Testing**

In system testing the entire software is tested . The reference document for this process is the requirement document and the goal is to see whether the software meets its requirements. The system was tested for various test cases with various inputs.

**Acceptance Testing**

Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactory. Testing here focus on the external behavior of the system , the internal logic of the program is not emphasized . In acceptance testing the system is tested for various inputs.

**Types of Testing**

1. Black box or functional testing
2. White box testing or structural testing

**Black box testing**

This method is used when knowledge of the specified function that a product has been designed to perform is known . The concept of black box is used to represent a system whose inside workings are not available to inspection . In a black box the test item is a "Black" , since its logic is unknown , all that is known is what goes in and what comes out , or the input and output.

Black box testing attempts to find errors in the following categories:

Incorrect or missing functions

Interface errors

Errors in data structure

Performance errors

Initialization and termination errors

As shown in the following figure of Black box testing , we are not thinking of the internal workings , just we think about

What is the output to our system?

What is the output for given input to our system?

**?**

Input Output

The Black box is an imaginary box that hides its internal workings

**White box testing**

White box testing is concerned with testing the implementation of the program. the intent of structural is not to exercise all the inputs or outputs but to exercise the different programming and data structure used in the program. Thus structural testing aims to achieve test cases that will force the desire coverage of different structures . Two types of path testing are statement testing coverage and branch testing coverage.

**INTERNAL WORKING**

Input Output

The White Box testing strategy, the internal workings

**Test Plan**

Testing process starts with a test plan. This plan identifies all the testing related activities that must be performed and specifies the schedules, allocates the resources, and specified guidelines for testing . During the testing of the unit the specified test cases are executed and the actual result compared with expected output. The final output of the testing phase is the test report and the error report.

**Test Data:**

Here all test cases that are used for the system testing are specified. The goal is to test the different functional requirements specified in Software Requirements Specifications (SRS) document.

**Unit Testing:**

Each individual module has been tested against the requirement with some test data.

**Test Report:**

The module is working properly provided the user has to enter information. All data entry forms have tested with specified test cases and all data entry forms are working properly.

**Error Report:**

If the user does not enter data in specified order then the user will be prompted with error messages. Error handling was done to handle the expected and unexpected errors.

**9.2 TEST CASES:**

Test cases can be divided in to two types. First one is Positive test cases and second one is negative test cases. In positive test cases are conducted by the developer intention is to get the output. In negative test cases are conducted by the developer intention is to don’t get the output.

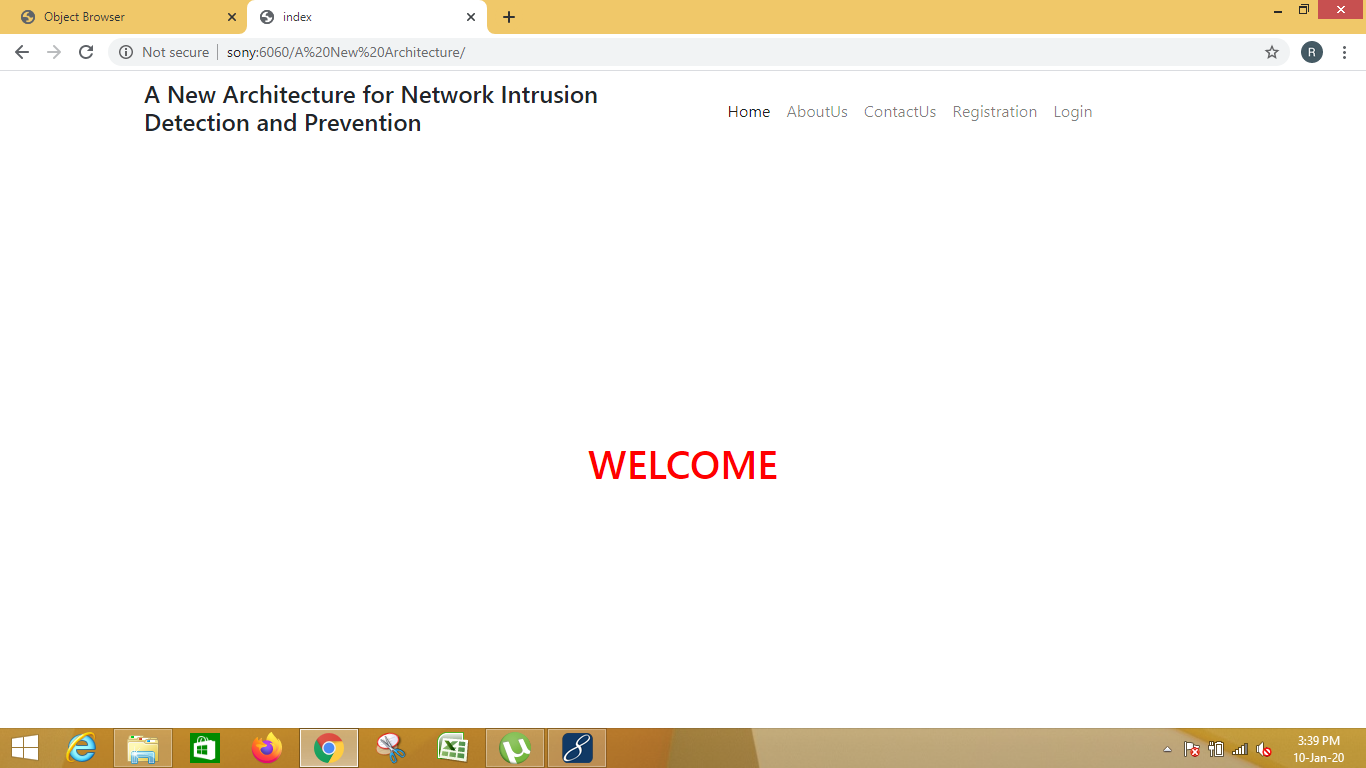
**+VE TEST CASES**

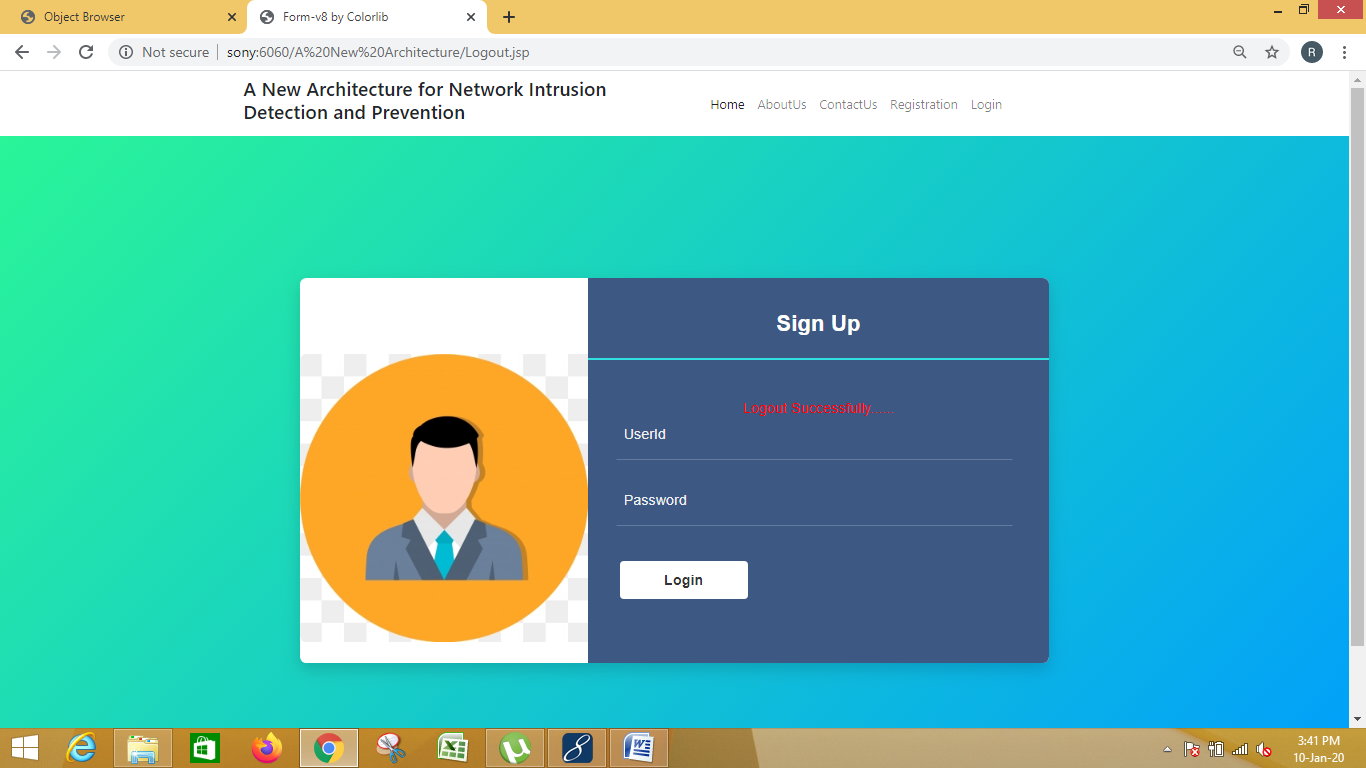
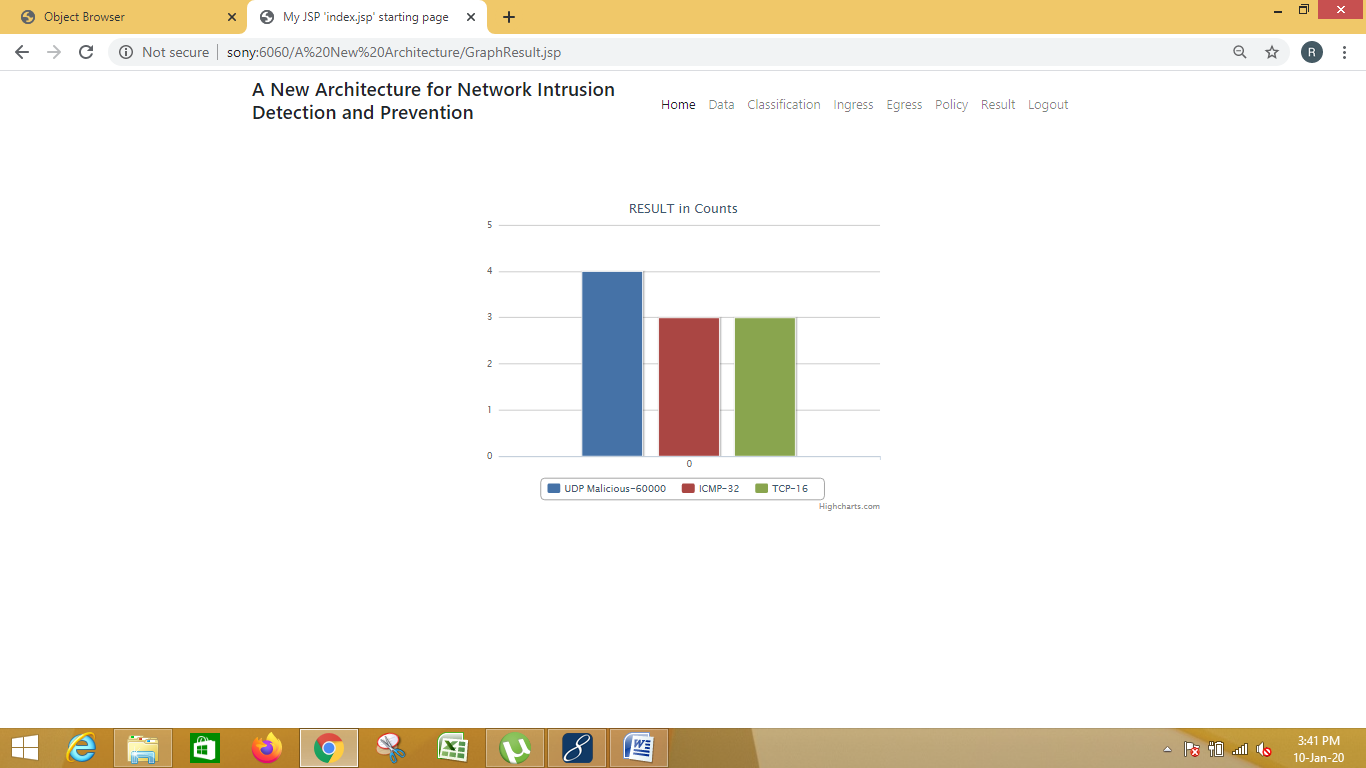
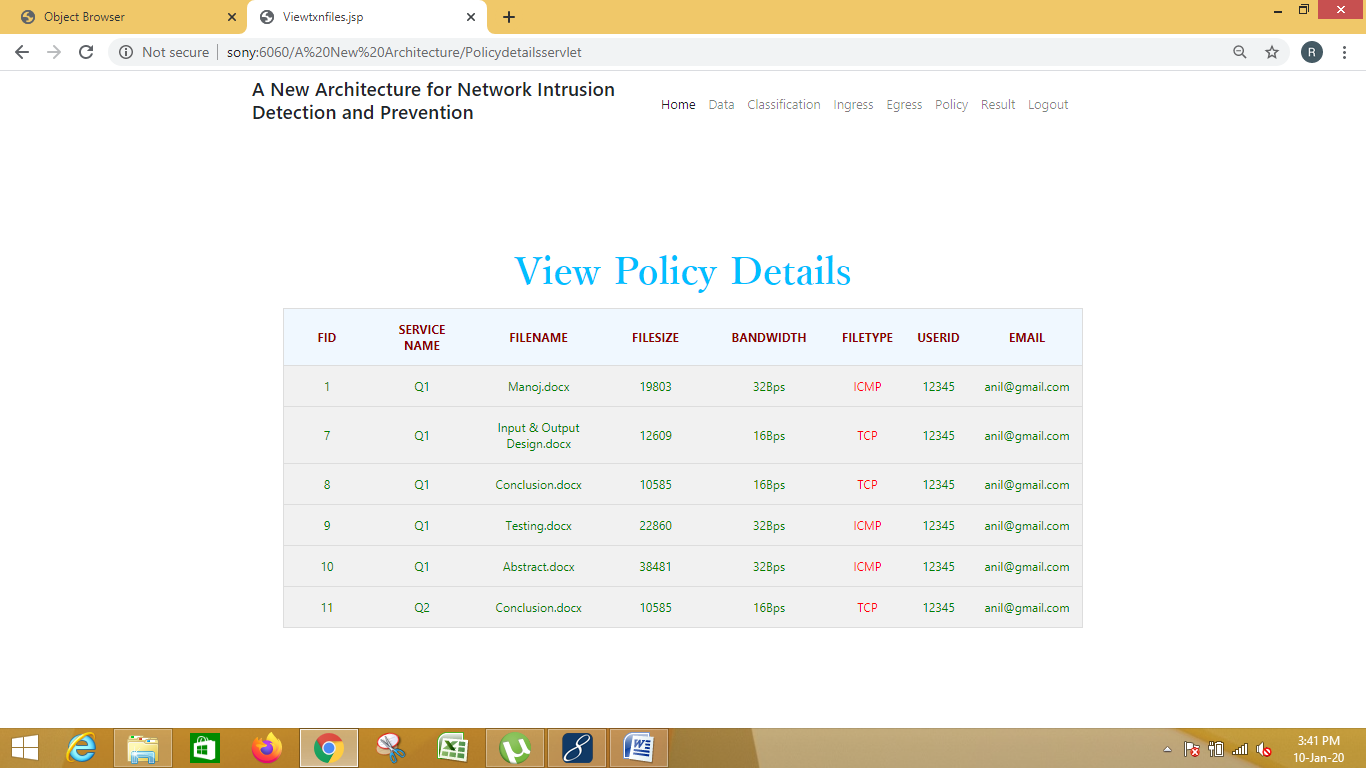
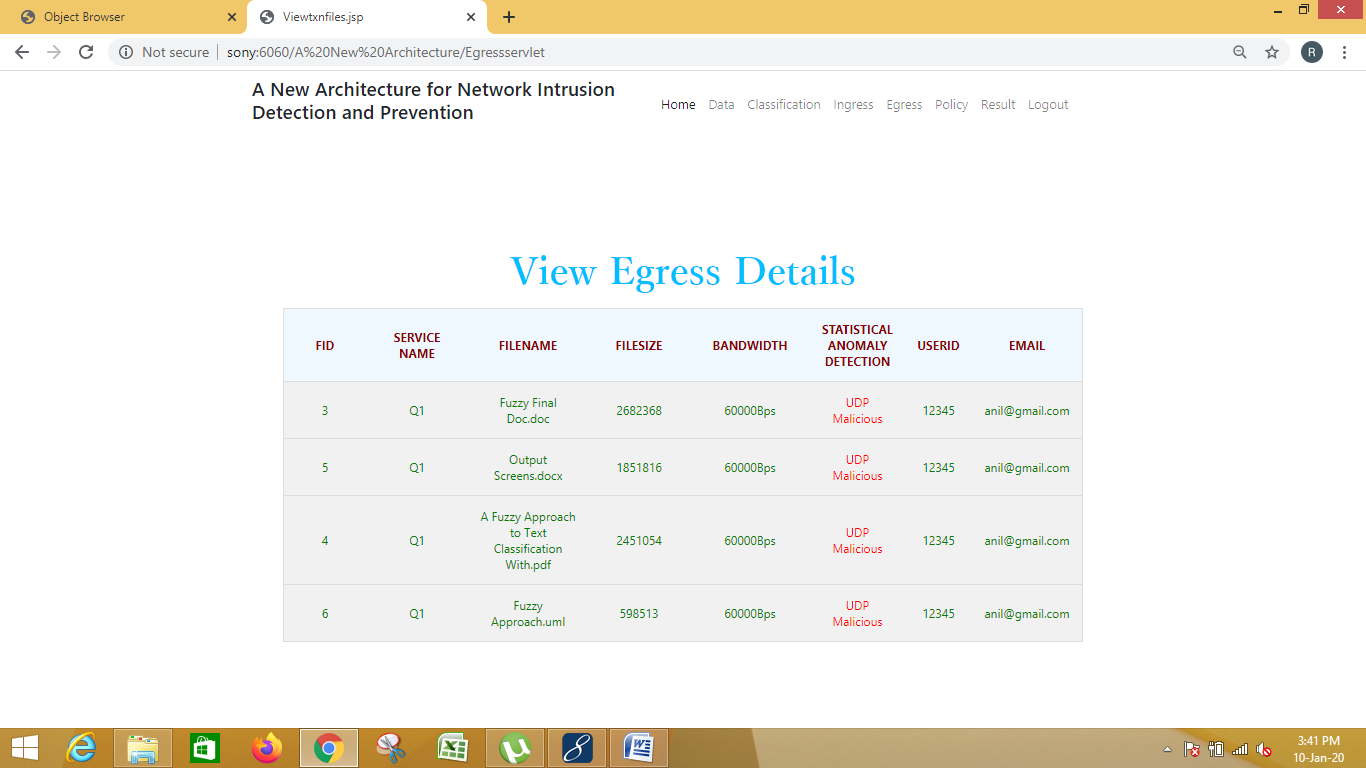
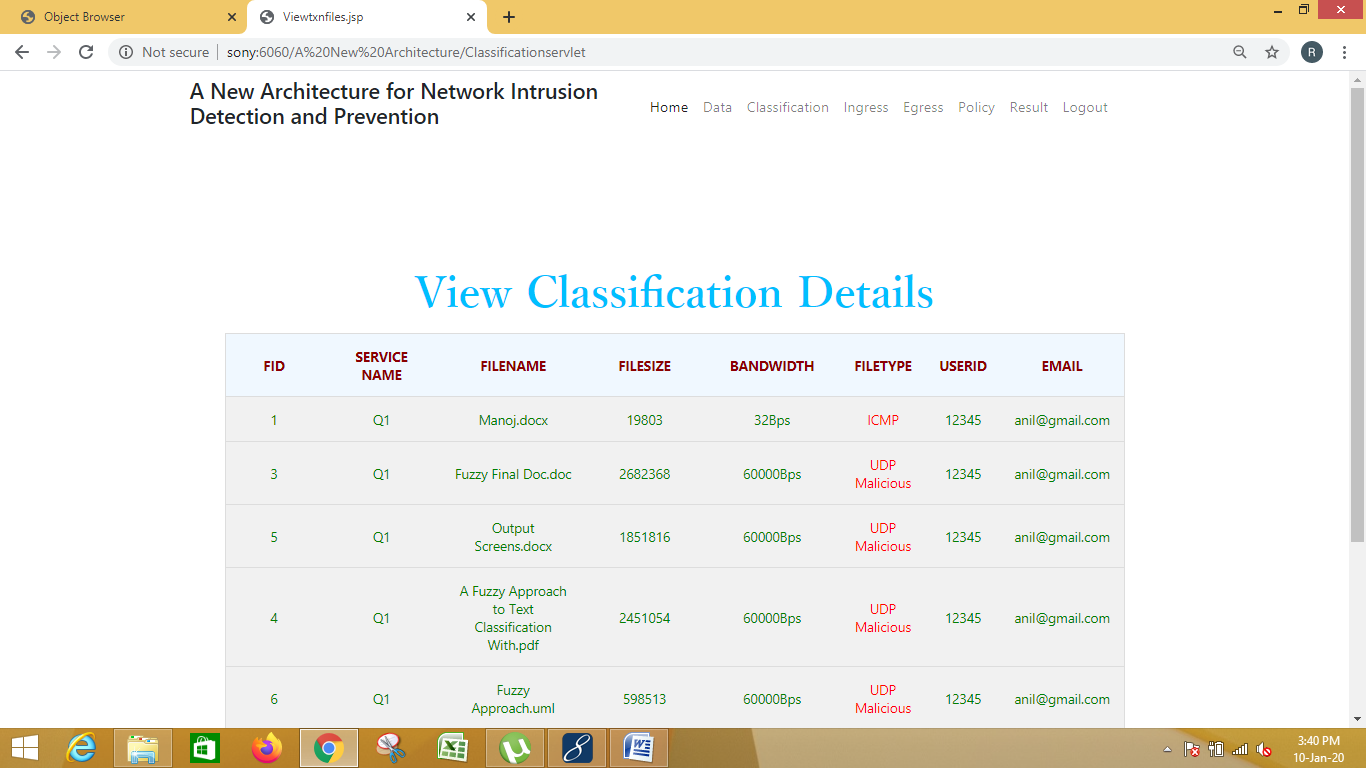
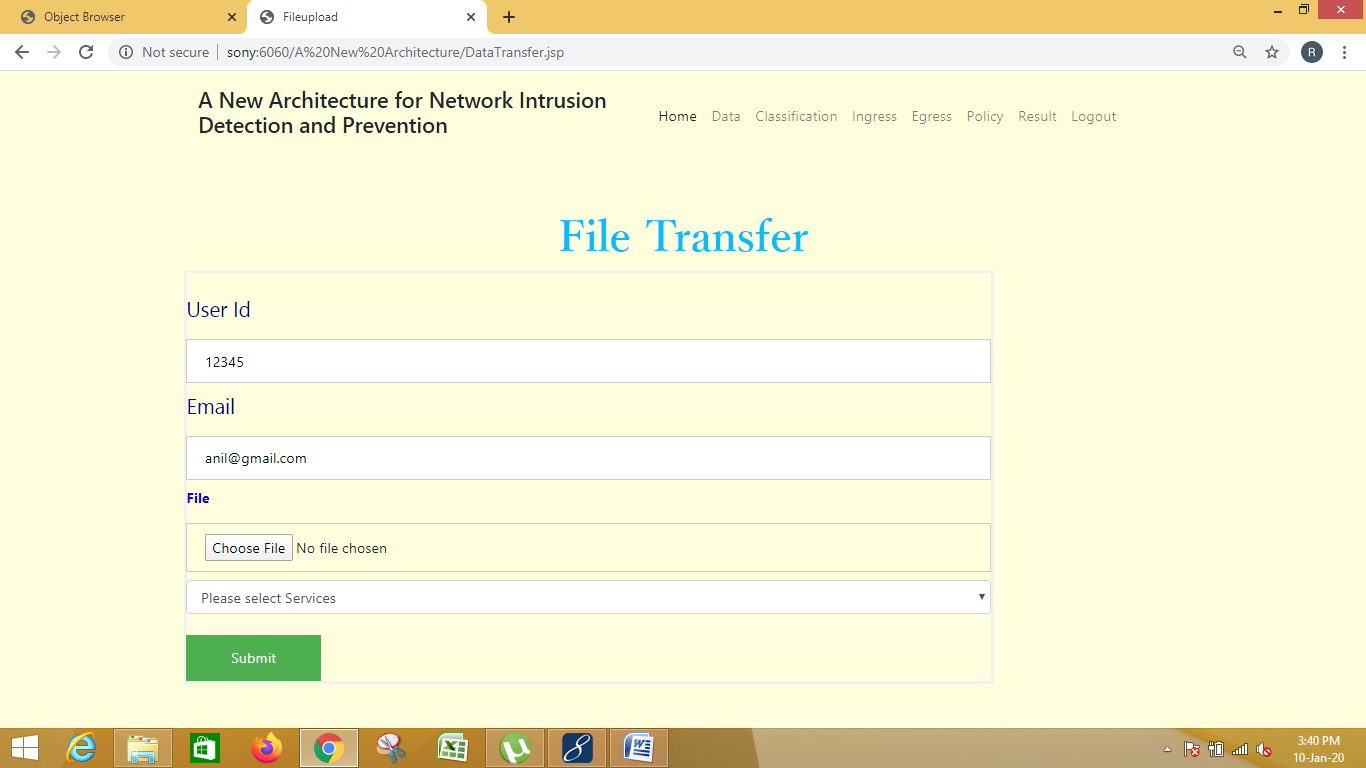
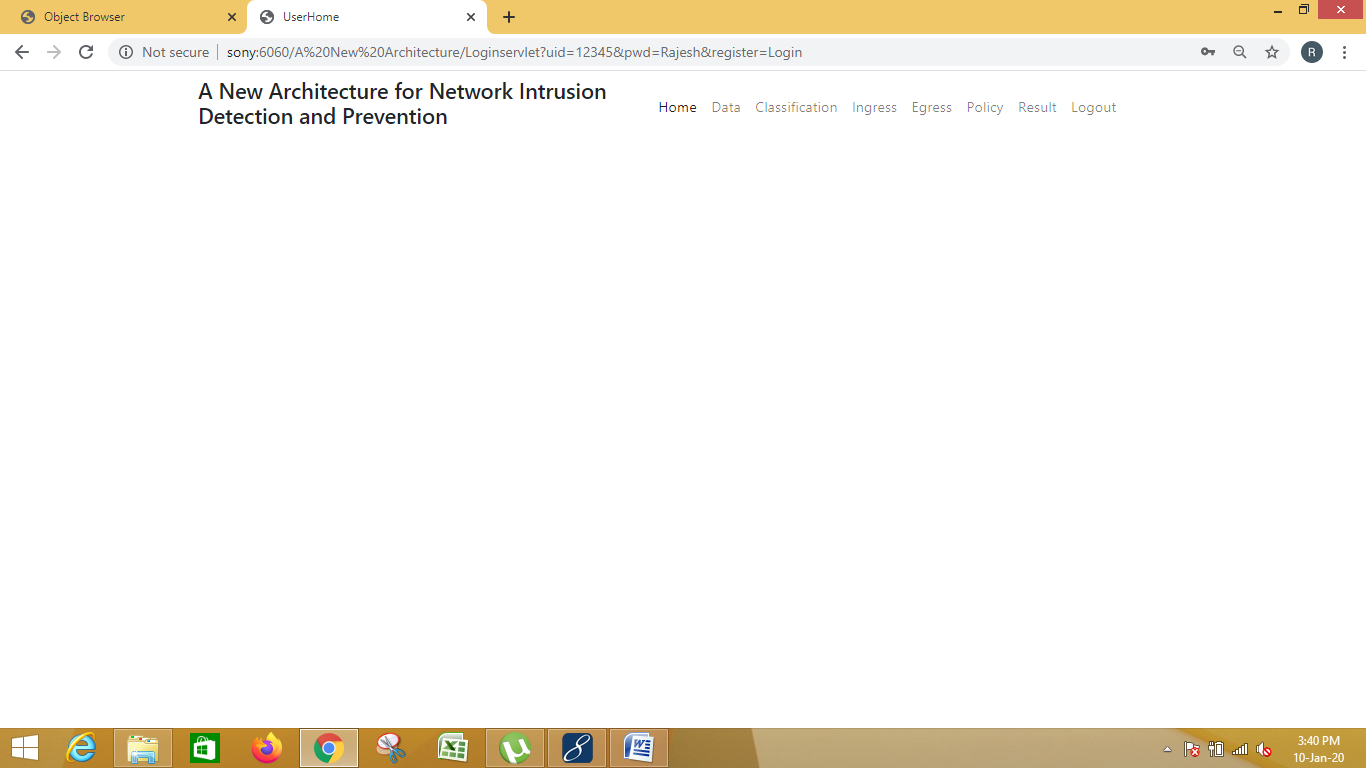
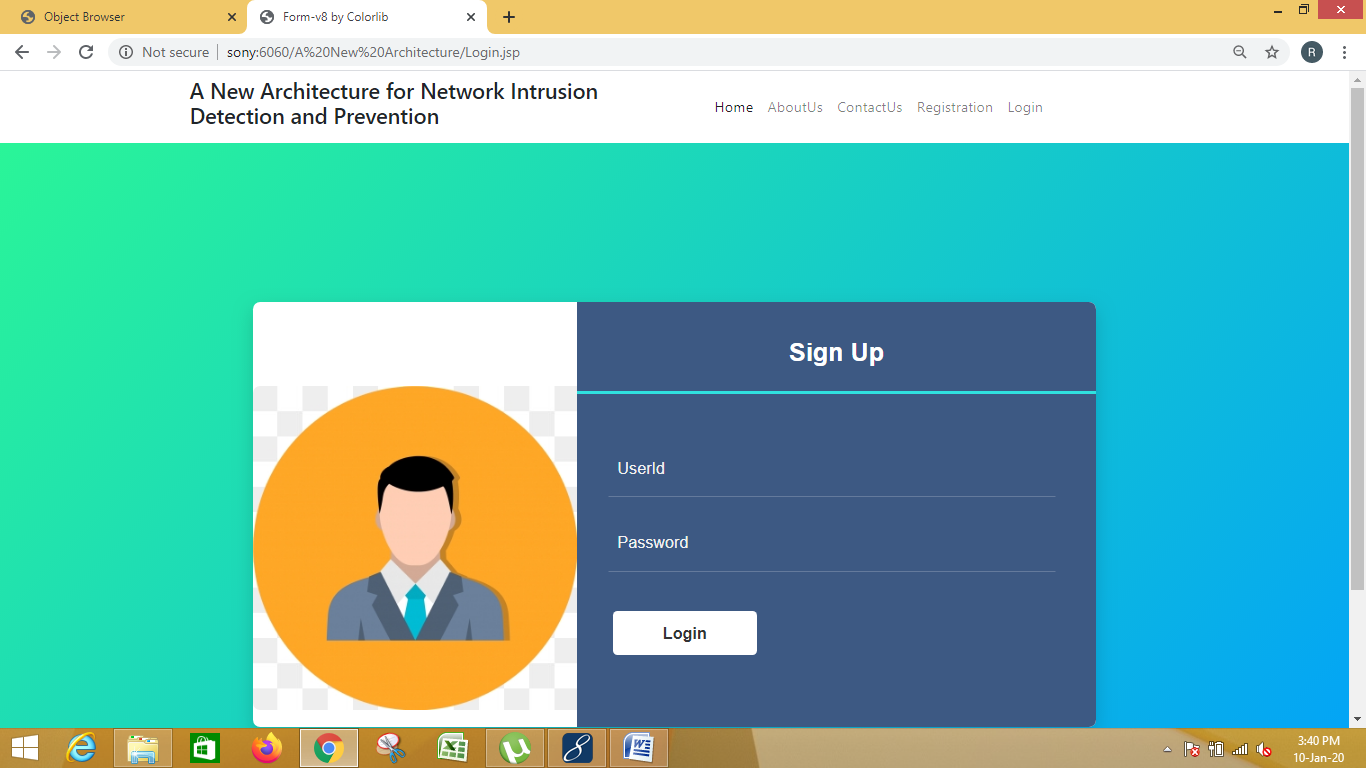
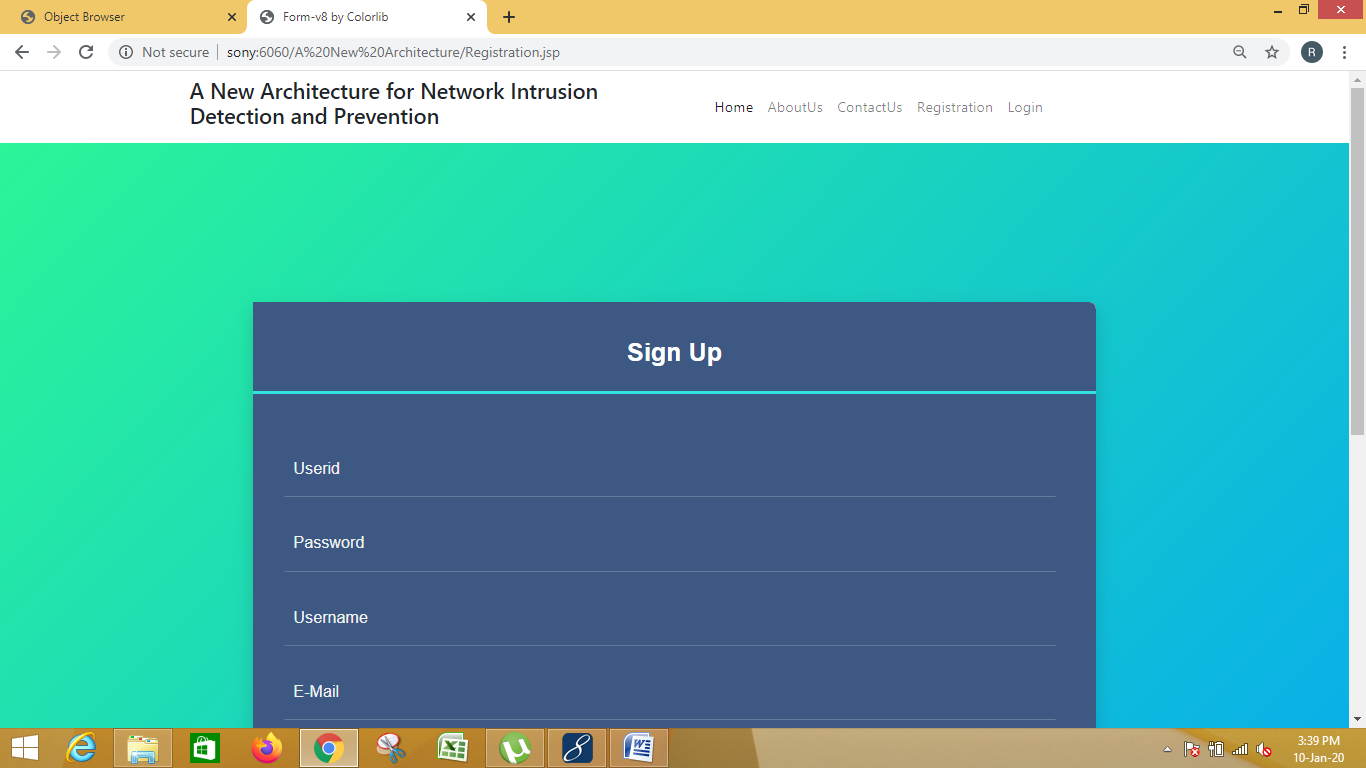
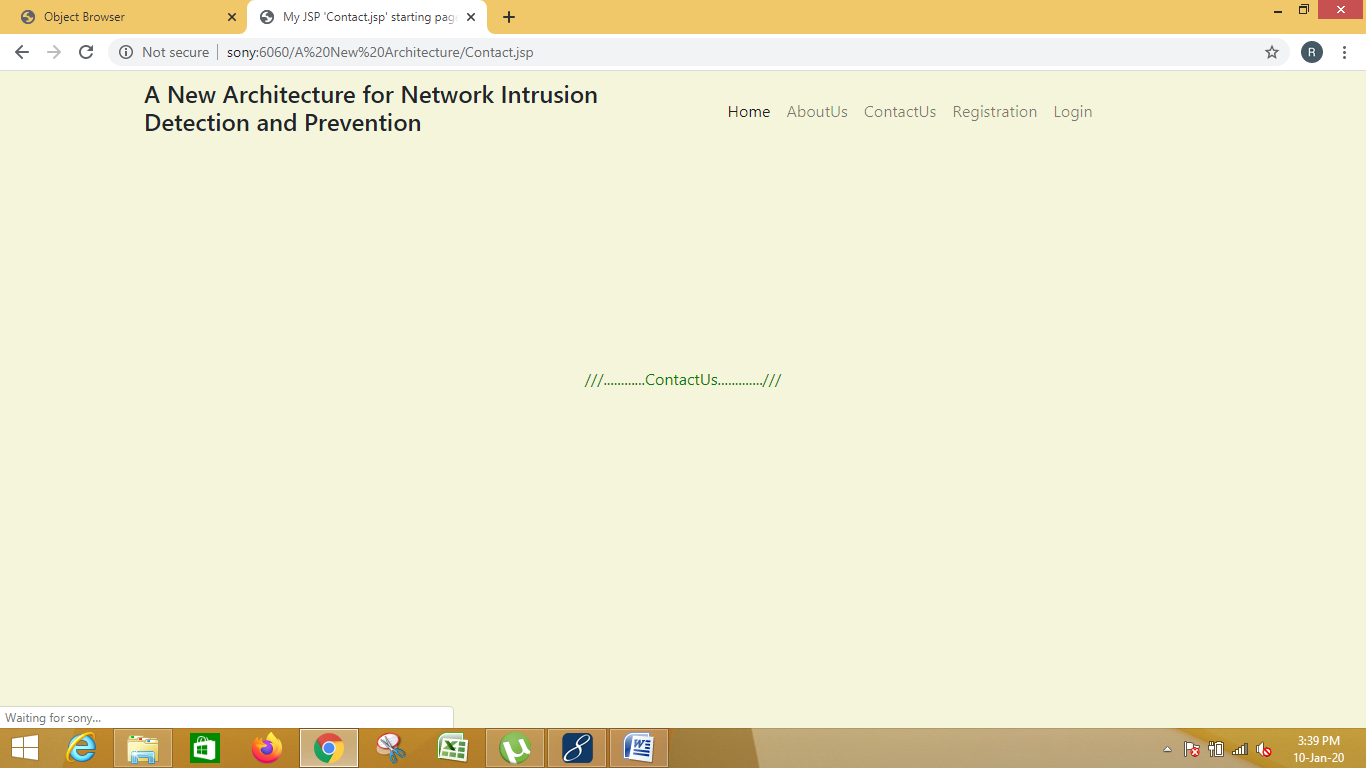
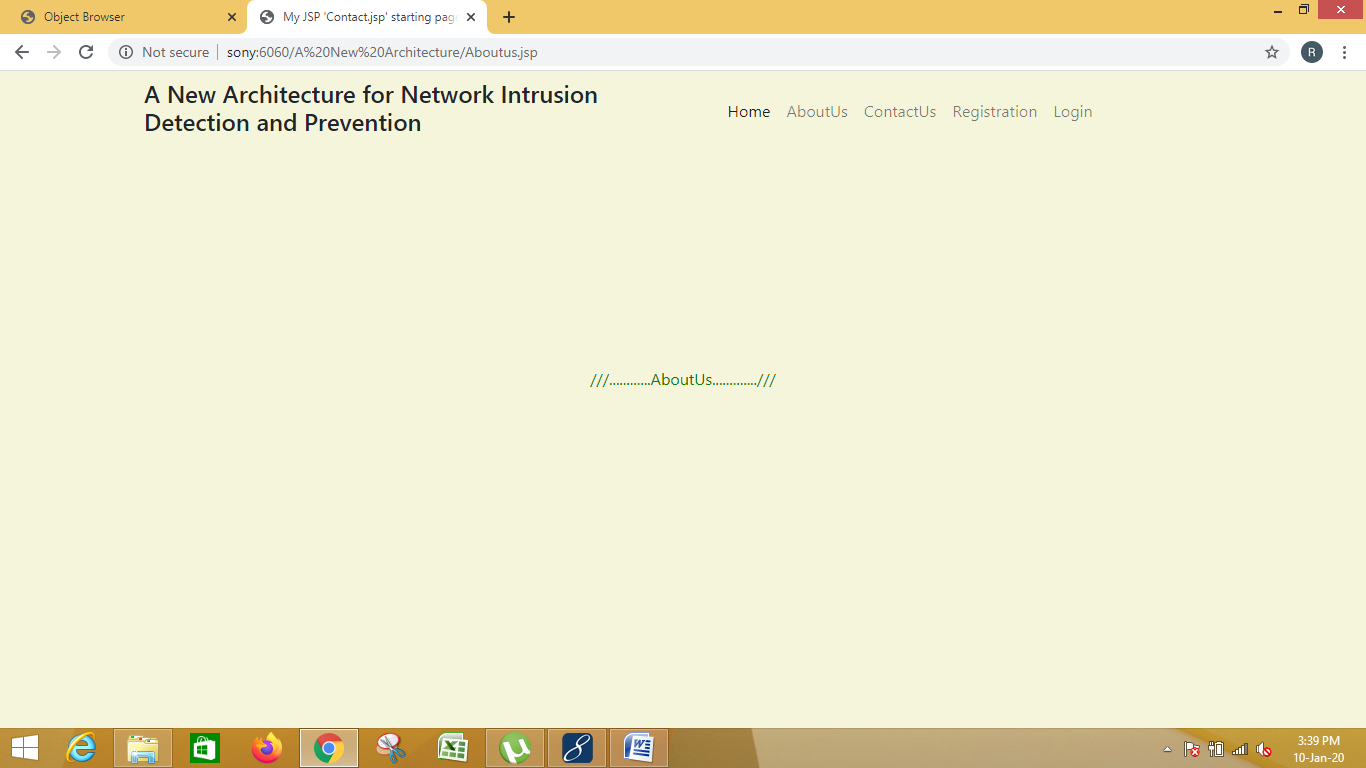
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Test case Description** | **Actual value** | **Expected value** | **Result** |
| **1** | Run source ,router ,destination window | Window open | Pop up window display | True |
| **2** | Browse file | Upload file | Data Stored into Database successfully | True |

**-VE TEST CASES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Test case Description** | **Actual value** | **Expected value** | **Result** |
| **1** | File browse | Upload file | Not Display uploaded content | False |

**10. Output Screens:**





**11. Future Enhancement:**

Statistical based anomaly detection is designed to detect deviations from a baseline model of network behavior. When the rate of ``malicious'' packet transmission is very high, the attack will almost certainly be detected by a statistical anomaly detector.

**12. Conclusion:**

A new architecture for NIDPS deployment was designed, implemented and evaluated. There has recently been massive development in computer networks regarding their ability to handle different speeds and data volumes. As a result of this rapid development, computer networks are now more vulnerable than ever to high-speed attacks and threats. These can cause considerable trouble to computer networks and systems. Network intrusions can be categorized at various levels. Many high-speed attacks can be classified as being difficult to detect or prevent. It will become ever more difficult to analyze increasing volumes of traffic due to the rapid shifts in technology that are increasing network speed.

**13. References:**

[1].B.Wang,Y.Zheng,W.Lou,andY.T.Hou,‘‘DDoS attack protection in the era of cloud computing and software-deﬁned networking,’’Comput.Netw., vol. 81, pp. 308–319, Mar. 2015.

[2] K. Chauhan and V. Prasad, ‘‘Distributed denial of service (DDoS) attack techniques and prevention on cloud environment,’’ Int. J. Innov. Advance- ment Comput. Sci., vol. 4, pp. 210–215, Sep. 2015.

[3] M. D. Samani, M. Karamta, J. Bhatia, and M. B. Potdar, ‘‘Intrusion detectionsystemforDoSattackincloud,’’InternationalJournalofApplied Information Systems (Foundation of Computer Science), vol. 10, no. 5. New York, NY, USA: FCS, 2016.

[4] S.H.Vasudeo,P.Patil,andR.V.Kumar,‘‘IMMIX-intrusiondetectionand prevention system,’’ in Proc. Int. Conf. Smart Technol. Manage. Comput., Commun., Controls, Energy Mater. (ICSTM), May 2015, pp. 96–101.

[5] W. Bul’ajoul, A. James, and M. Pannu, ‘‘Improving network intrusion detectionsystemperformancethroughqualityofserviceconﬁgurationand parallel technology,’’ J. Comput. Syst. Sci., vol. 81, no. 6, pp. 981–999, 2015.

[6] N.Akhtar,I.Matta,andY.Wang,‘‘ManagingNFVusingSDNandcontrol theory,’’ Dept. CS, Boston Univ., Boston, MA, USA, Tech. Rep. BUCS- TR-2015-013, 2015.

[7] P. S. Kenkre, A. Pai, and L. Colaco, ‘‘Real time intrusion detection and prevention system,’’ in Proc. 3rd Int. Conf. Frontiers Intell. Comput., Theory Appl. (FICTA). Bhubaneswar, India: Springer, 2015, pp. 405–411.

[8] M. Li, J. Deng, L. Liu, Y. Long, and Z. Shen, ‘‘Evacuation simulation and evaluation of different scenarios based on trafﬁc grid model and high performance computing,’’ Int. Rev. Spatial Planning Sustain. Develop., vol. 3, no. 3, pp. 4–15, 2015.

[9] J.-M. Kim, A.-Y. Kim, J.-S. Yuk, and H.-K. Jung, ‘‘A study on wireless intrusion prevention system based on snort,’’ Int. J. Softw. Eng. Appl., vol. 9, no. 2, pp. 1–12, 2015.

[10] Cisco. (2016). Cisco Interfaces and Modules, Cisco Security Mod- ules for Security Appliances. Accessed: Feb. 30, 2018. [Online]. Avail- able: http://www.cisco.com/c/en/us/support/interfaces-modules/security-modules-security-appliances/tsd-products-support-series-home.html

[11] M. Trevisan, A. Finamore, M. Mellia, M. Munafò, and D. Rossi, ‘‘DPD- KStat: 40Gbps statistical trafﬁc analysis with off-the-shelf hardware,’’ Telecom, Paris, France, Tech. Rep. 318627, 2016.

[12] W. Bul’ajoul, A. James, S. Shaikh, and M. Pannu, ‘‘Using Cisco network components to improve NIDPS performance,’’ Comput. Sci. Inf. Technol., pp. 137–157, Aug. 2016.

[13].K.R.Kishore,A.Hendel,andM.V.Kalkunte,‘‘System,methodandappa-ratus for network congestion management and network resource isolation,’’ U.S. Patent 9762497, Sep. 12, 2017.

[14] Y. Naouri, and R. Perlman, (2015). ‘‘Network congestion management by packet circulation,’’ U.S. Patent8989017B2, Mar. 24, 2015.

[15] Y. Zhu et al., ‘‘Packet-level telemetry in large datacenter networks,’’ in Proc. ACM Conf. Special Interest Group Data Commun. New York, NY, USA: ACM, 2015, pp. 479–491.

[16] T. Szigeti, C. Hattingh, R. Barton, and K. Briley, Jr., End-to-End QoS Network Design: Quality of Service for Rich-Media & Cloud Networks. London, U.K.: Pearson Education, 2013.

[17] M. K. Testicioglu and S. K. Keith, ‘‘Method for prioritizing network packets at high bandwidth speeds,’’ U.S. Patent15804940, Nov. 6, 2017.

[18] T. Szigeti, J. Henry, and F. Baker, Mapping Diffserv to IEEE 802.11 Yes, Tatil, document RFC 8325, 2018.

[19] D. Melman, I. Mayer-Wolf, C. Arad, and R. Zemach, ‘‘Egress ﬂow mir- roring in a network device,’’ U.S. Patent15599199, May 18, 2017.

[20] K. K. Kulkarni, and R. O. Nambiar, ‘‘Distributed application framework for prioritizing network trafﬁc using application priority awareness,’’ U.S. Patent15792635, Oct. 24, 2017.

**14. Bibliography:**

(1) Java Complete Reference by Herbert Shield

(2) Database Programming with JDBC and Java by George Reese

(3) Java and XML By Brett McLaughlin

(4) Wikipedia, URL: <http://www.wikipedia.org>.

(5) Answers.com, Online Dictionary, Encyclopedia and much more, URL: <http://www.answers.com>

(6) Google, URL: <http://www.google.co.in>

(7)Project Management URL: <http://www.startwright.com/project.html>