SMARTECH

NetCheck Project
Detailed Design Report

Neslihan Bulut

Kezban Demirtaş

Hande Çelikkanat

Gülşah Karaduman

Filiz Alaca

Department of Computer Engineering

METU

January 2006

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Purpose of this Document

The purpose of this document is to elaborate the design specifications of the project. In this report, we intend to give detailed information about how our solutions fulfill the problem requirements. During our studies on this report, we have developed our sight to the problem and to the solution. We will present our project's modular specification and UML diagrams (use-case, class, sequence and activity diagrams) through which our understanding of the system improves.

1.2 Scope and Definition of the Project

In today's world, Internet has become the key tool in every aspect of life. With the increasing internet usage and the vulnerability of Internet to abuse, security gains more and more importance. Organizations are one of the areas where Internet is heavily used. In order not to lose confidential information about the company, about the projects they are working on, and mainly to ensure security policies, organizations have to take precautions for abuse. At this point, security tools appear to act as the protector against malicious usage.

NetCheck will be a web-based application level gateway which offers secure Internet access for the organizations. Our intended software 'NetCheck' will mainly provide the following facilities:

- Real-time network monitoring
- Content filtering
- Download restriction
- Access restriction
- Statistical data about network traffic
- Web interface for the control of the program
- Confidential data hiding

1.3 Project Overview

Administrative Facilities: The administrator of the system can define access and download rights for individuals in the local network via the web interface. He/she can define black words and black word groups, black URL and black URL groups, and also he/she can define user groups and assign black URL groups and black word groups to user groups. Furthermore, the administrator may specify confidential data of the organization in order to hide that data from outside world.

Network Traffic Monitoring: All incoming and outgoing web traffic will be displayed on a web page in real time. Source address, destination address, accessed URL, size of communication packets, and time of communication should be monitored.

Access Restriction: Restriction will be applied to individuals according to black URL groups which are assigned to the user. Also, in some time intervals, some specified URL's can be restricted to the user (e.g. URL x cannot be accessed between 09:00-11:00).

Download Restriction: The administrator will be able to specify a bandwidth limit for the users' download operation.

Content Filtering: Since in today's fast changing world, many sites are coming up continuously, a restriction mechanism that relies barely on blocking specific URLs can hardly provide satisfying results. So, the administrator is given the option of specifying a description of the content of sites that will not be allowed to the user. The administrator will have the option of choosing between two options. One is a Bayesian algorithm that executes by learning from inspected site contents, and another is a percentage-controlled algorithm that runs by counting the occurrences of black and white words in the incoming packets. These will allow the administrator to choose the more appropriate one for his / her organization, keeping the control rights in his/her hands or letting the system run autonomously.

2 SYSTEM MODULES

2.1 System Management Module

Aim of the Module:

This module will provide the administrator with a web-based interface to manage the system. Firstly, the administrator will be asked for his/her username and password for the authentication and after the username/password verification, the administrator will have the right to control or monitor the following system features via that interface;

- Monitoring network traffic,
- Defining local IP's for the system
- Defining access and download rights for local IP's,
- Restricting URL access,
- Specifying black & white word lists,
- Specifying the running mode of the program,
- Specifying confidential data for the organization,
- Monitoring network statistics.

General Description and Interactions with the System:

System Management Module will interact with the system via the database.

To enable the administrator for monitoring the network traffic it will interact with the network traffic monitoring module and collect the corresponding data, namely source IP, destination IP, URL, connection size and time from the NetworkTrafficLog table.

For specifying access and download rights of the users this module is going to get the input data from the administrator through web interface and update the corresponding tables. For inserting a new user to the system, administrator should define the IP, black words, black word groups, black URL, black URL groups and the personal information about the user. This part will be updating User, UserInGroup, ExtraURL, ExtraURLGroup, ExtraWord and ExtraWordGroup tables in the database.

URL's that are defined as black by the administrator will be added to the BlackURLList table, also specified groups for the black URLs will be inserted to URLInGroup table. In addition, time interval restriction for URL blocking will also be handled via the updates on the BlackURL table. Black and white word lists are going to be inserted into the BlackWord and WhiteWord tables.

Auto update of words and URLs are going to be handled by parsing the newly coming URLs and phrases from predefined sites such as urlblacklist.com. Consequently, BlackURL, BlackWord, WhiteURL and WhiteWord tables will be updated. Black/white URL and word lists are specified in the download sites as phrases on each line, which are going to be parsed in accordance.

Statistics display will be handled by requesting statistic data from Statistics module; it will be visualized in a user friendly manner.

For efficiency reasons our system has 3 different running modes, namely;

- ***Free Mode:*** The free mode will only provide monitoring network traffic and statistical data related to the network traffic. There will be no filtering or restriction mechanism.
- ***Normal Mode:*** This mode will be the default mode of the system. In addition to the network traffic monitoring and statistics, this mode will support URL access and download size restriction, content filtering and logging.
- ***Secure Mode:*** In addition to normal mode facilities, this mode will include a mechanism for providing security for confidential data such as a formula invented by a company working for pharmaceutics.

Specified running mode will be kept in the database and will trigger the modules associated with the mode.

Confidential data must be entered with the corresponding criticality of the keyword. These are stored in the ConfidentialData table.

Intended Procedure To Be Followed:

This module provides some authentication control features for preventing unauthorized people to access the program. To prevent unauthorized users from accessing the program by simply using the computer of someone who does have access, the system will log users out after a specified time of inactivity. Also to prevent brute force password attacks, delay on continuing failed login attempts from the same host will be expanded and hosts that have a given number of failed login attempts will be blocked.

After authentication to the system, the administrator can give orders and manage the system.

Input / Output Specifications:

Input is gathered from web interface with the proper use of html forms, and output will both be visualization on the web page and an associated update on the database.

Sample Views of the Module:

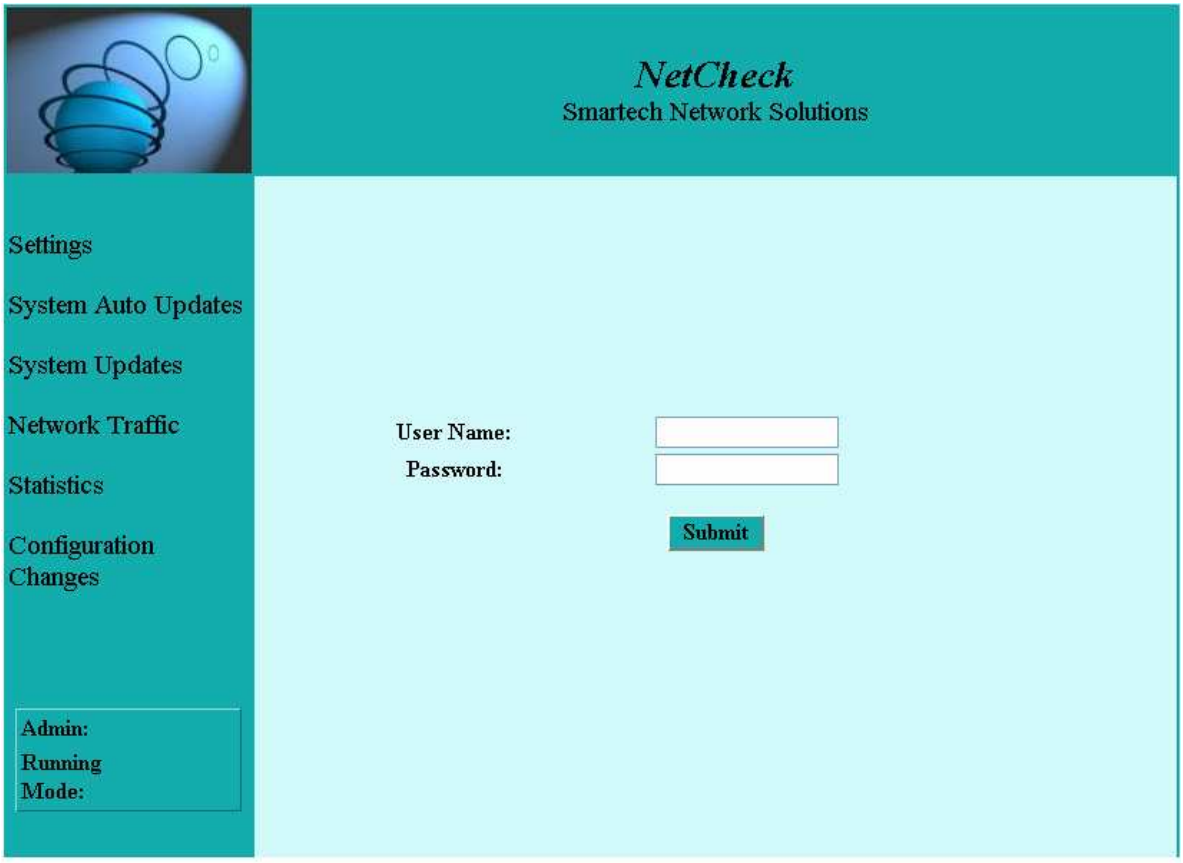


Figure-1: Login Screen

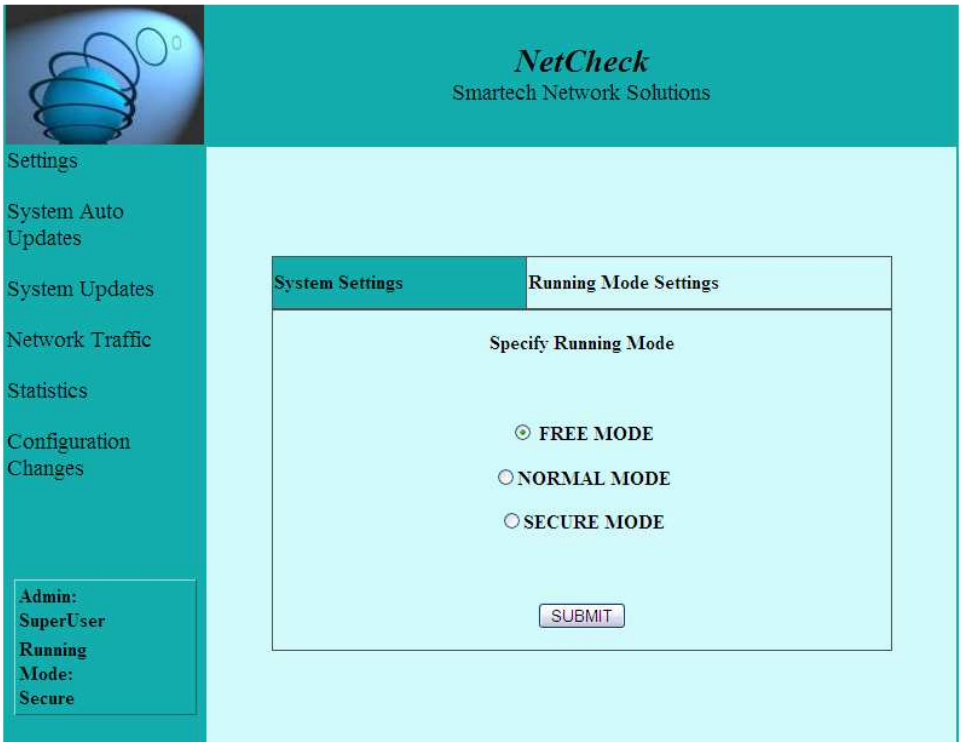


Figure-2: Running Mode Specification Screen

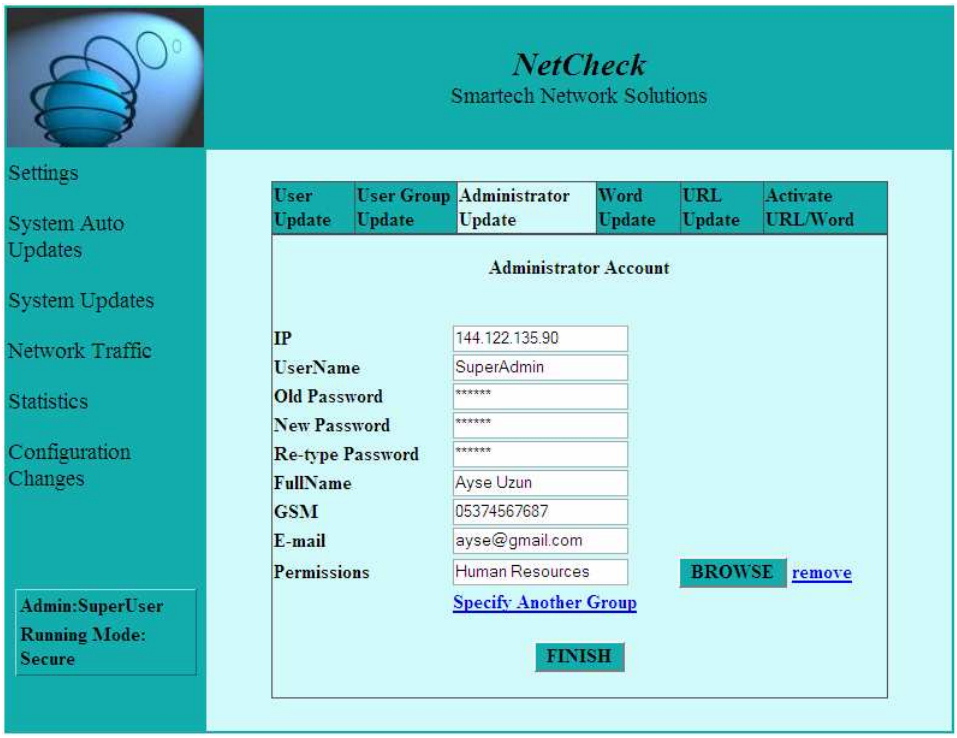


Figure-3: Administrator Update Screen

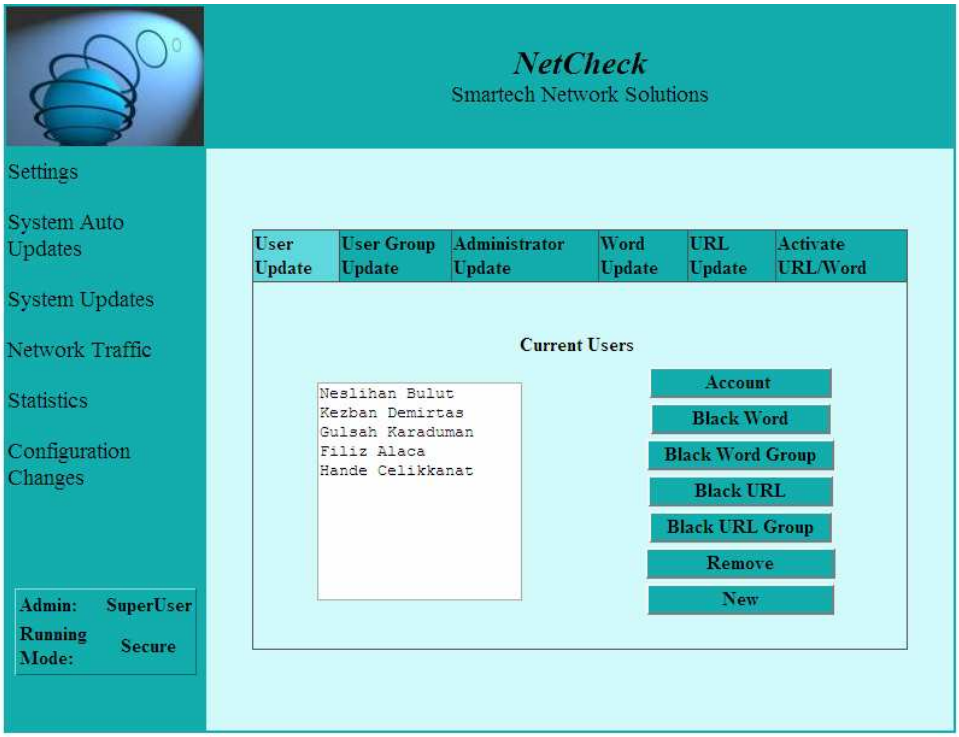


Figure-4: User Update Screen

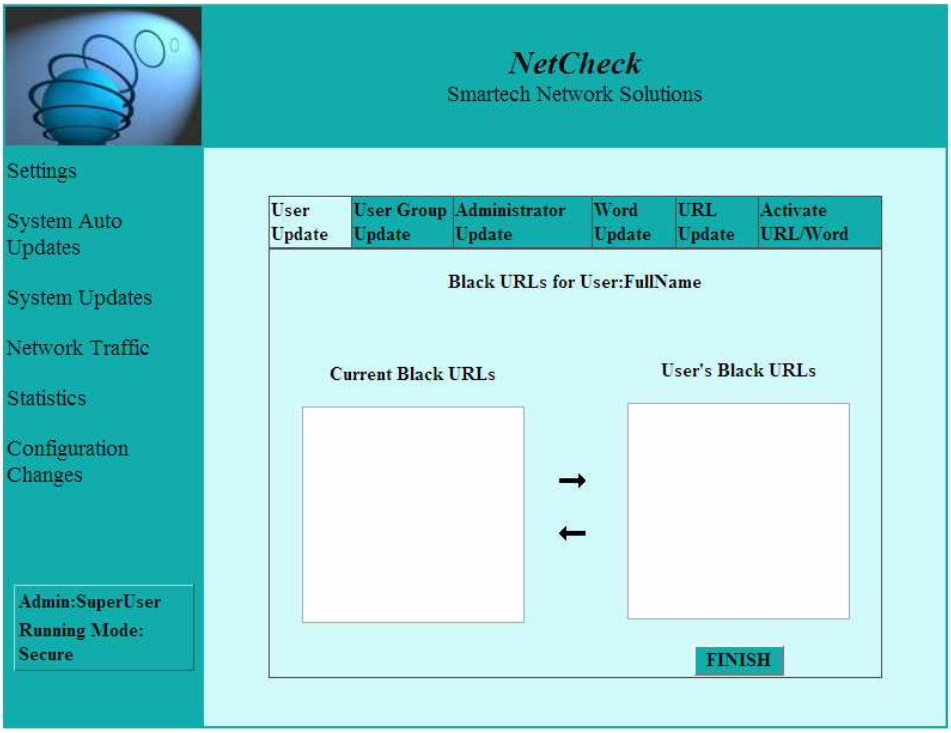


Figure-5: Black URL Specification for Users Screen

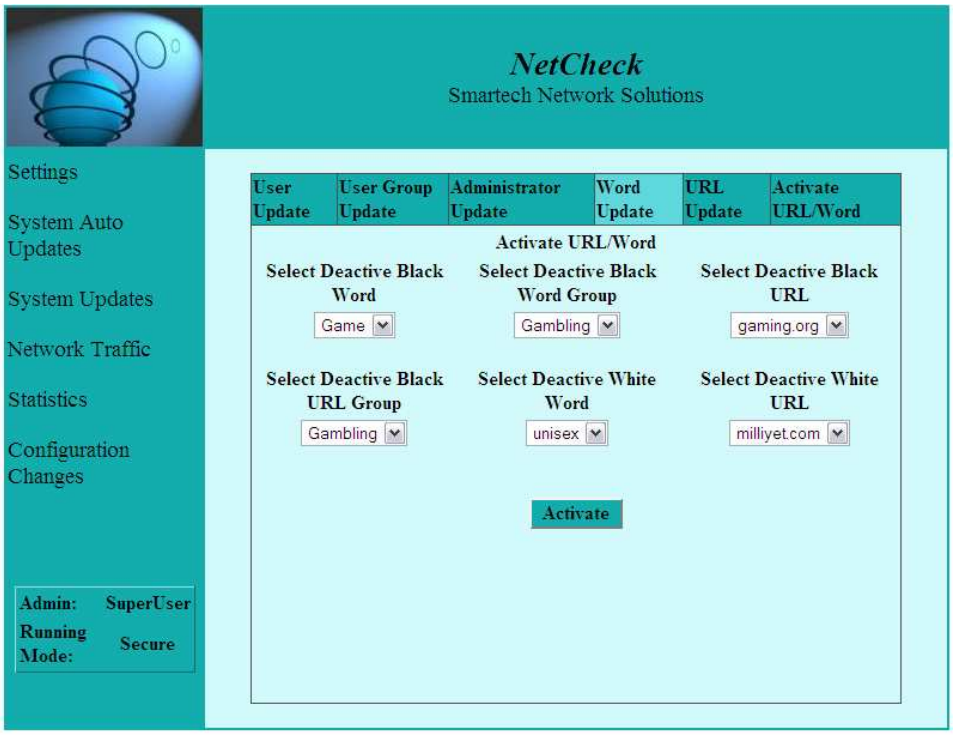


Figure-6: Black URL/Word Activation Screen

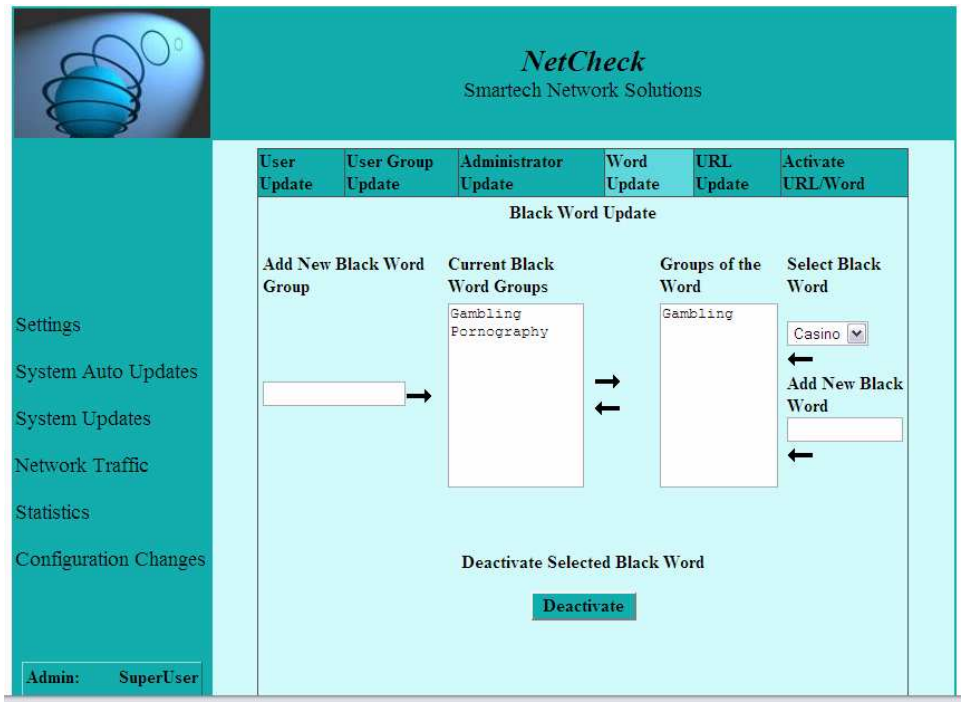


Figure-7: Black Word Update Screen

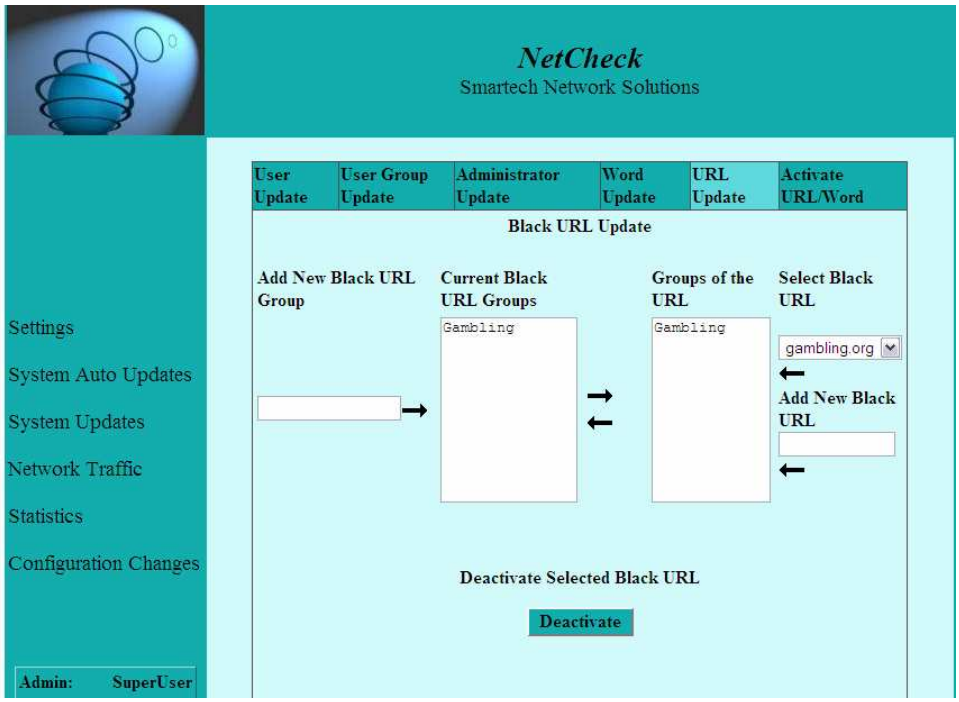


Figure-8: Black URL Update Screen

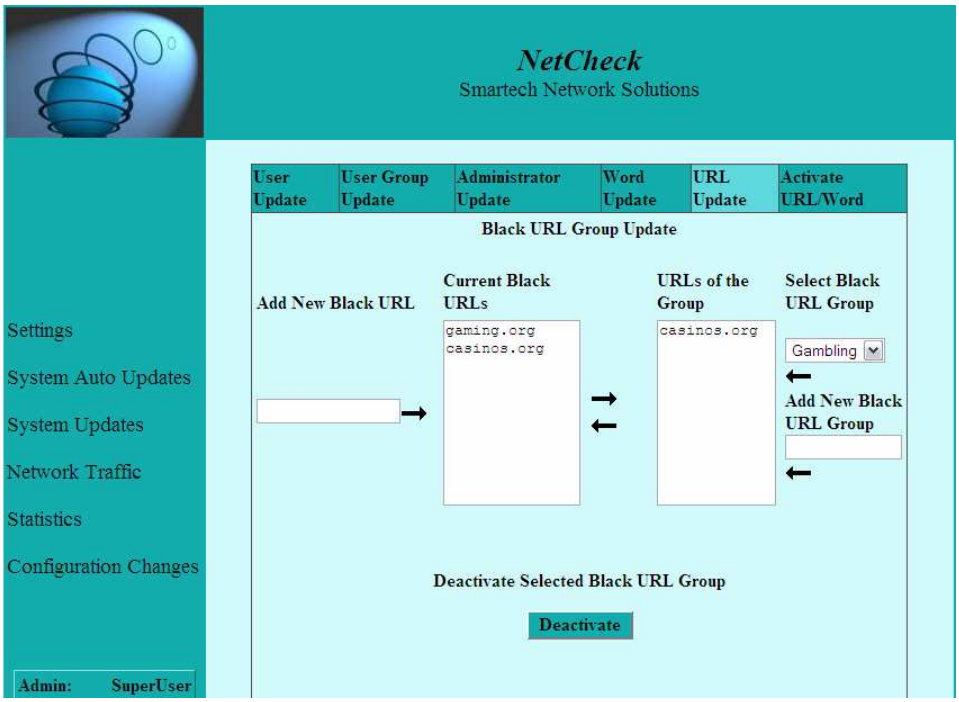


Figure-9: Black URL Group Update Screen

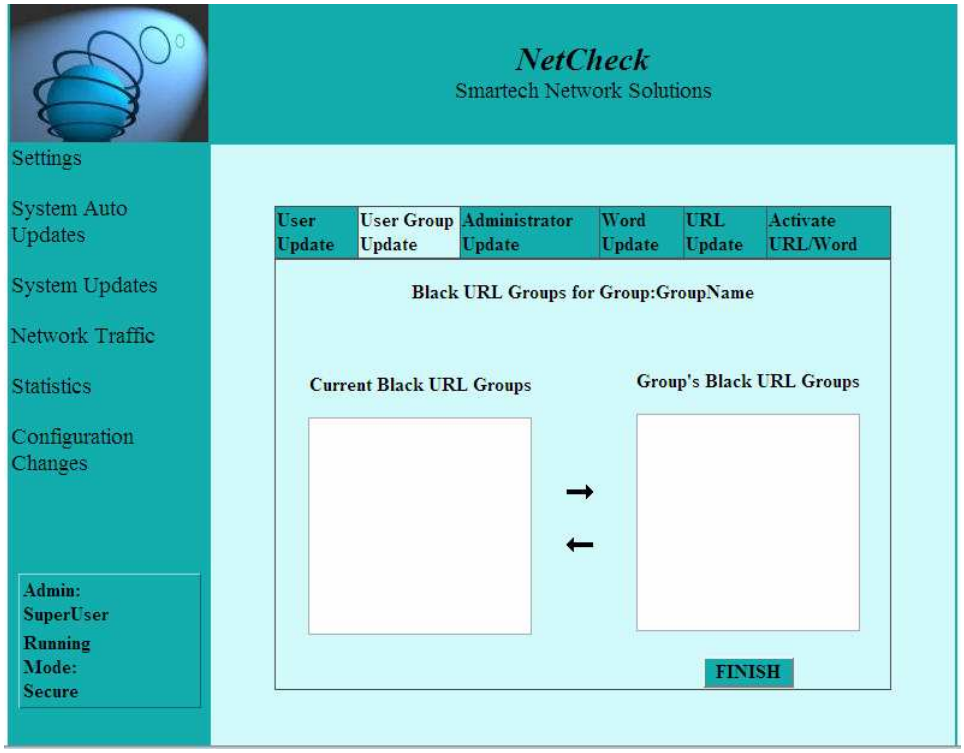


Figure-10: Black URL Group Specification for User Group Screen

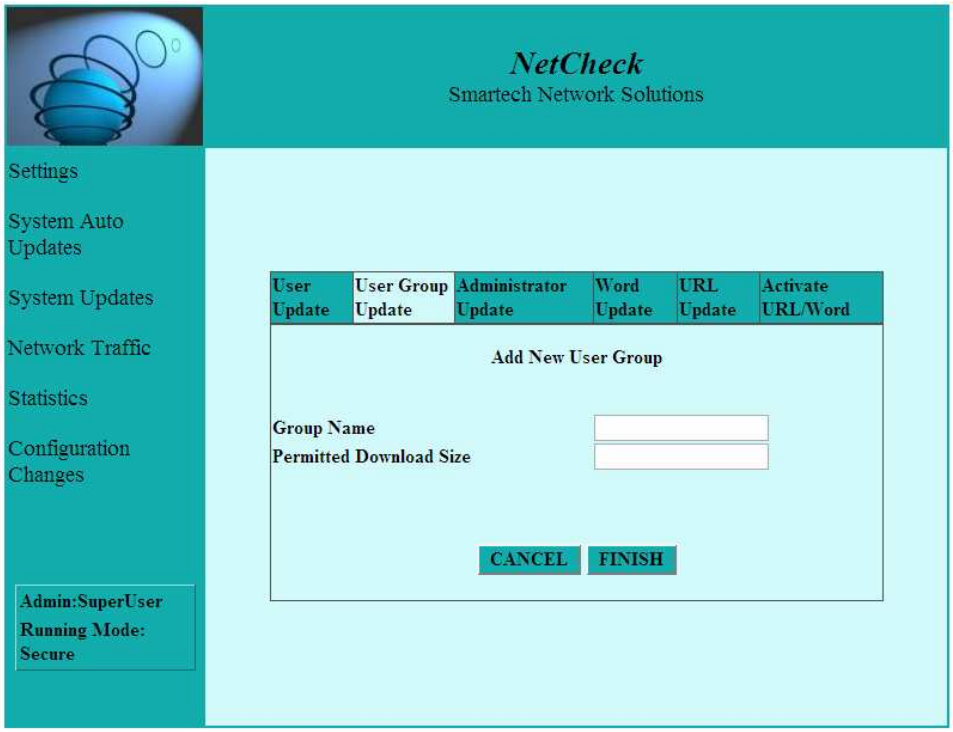


Figure-11: Add New User Group Screen

2.2 Network Traffic Monitoring Module

Aim of the Module:

This module will be responsible for monitoring incoming and outgoing network traffic in real time. Administrator will be informed about the source address, destination address, accessed URL, and size and time of the communication.

General Description and Interactions with the System:

This module will have a multi-threaded structure. It will be consisting of two threads, namely, Port Interaction and Database Interaction threads. The Port Interaction thread will be listening for incoming packets, parsing them and forwarding size and URL information to the Restriction Module. The Database thread will be saving the parsed information to the NetworkTrafficLog table in the database. This module will also be responsible for the address translation and routing of the packets.

Intended Procedure To Be Followed:

The packets caught by the Port Interaction thread will be inspected to decide if they are associated with the port numbered 80. In case they are, the packet will be parsed; acquiring source and destination IP's from IP packet header and the accessed URL from data part. Time and size information will be provided together with these to the Database thread, which provides the connections and updates of the database.

Input / Output Specifications:

The module will be getting TCP/IP packets from the network as input. As output, the module will be creating objects of packets for the rest of the system and also forwarding size and URL information to the Restriction Module.

Sample View of the Module:

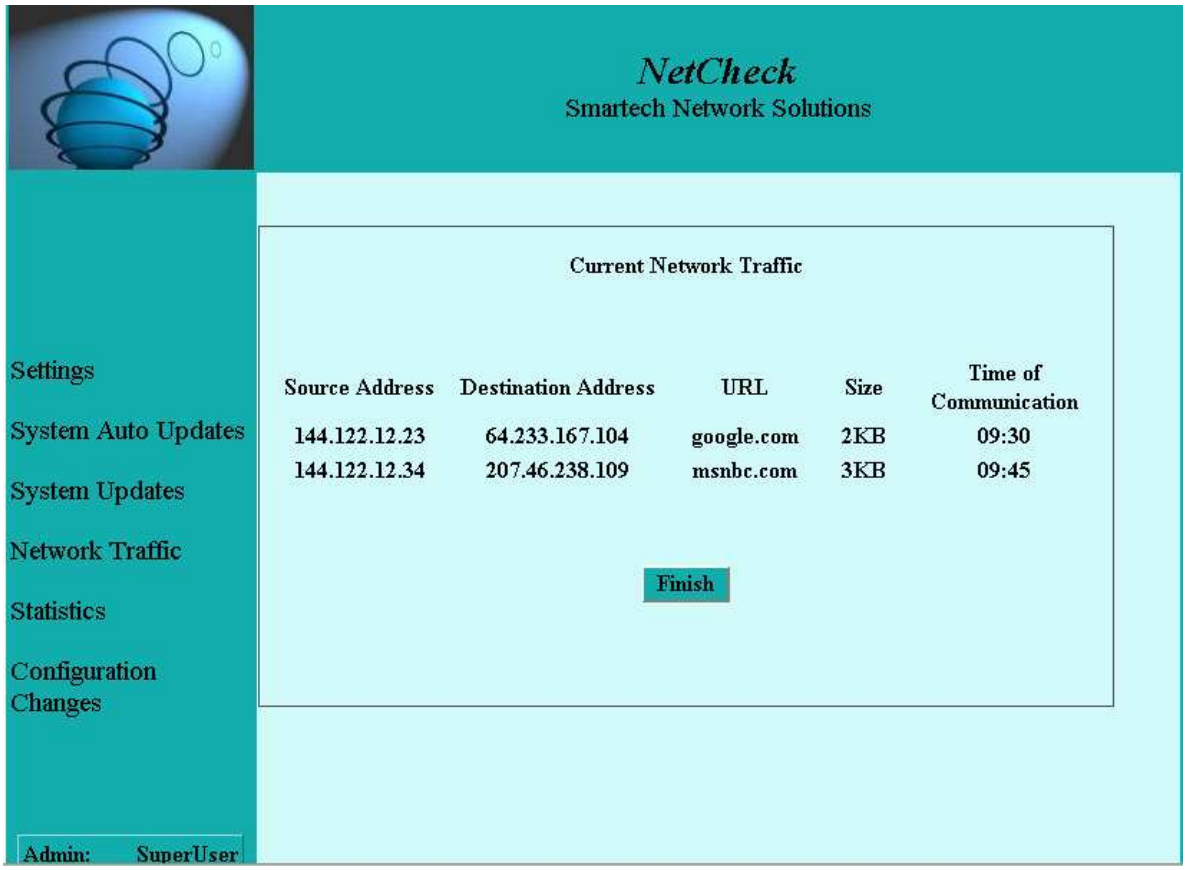


Figure-12: Monitoring Current Network Traffic Screen

2.3 Content Filtering Module

Aim of the Module:

This module will be responsible for filtering incoming packets in order to prevent malicious content coming from remote servers to the local area network. It will also filter outgoing packets to protect organizations' confidential information.

General Description and Interactions with the System:

This module operates on packets caught by the Network Traffic Monitoring Module. It distinguishes the incoming and outgoing packets with the help of source and destination IP information in the packets.

For incoming packets, a content filtering algorithm will be applied. The administrator of the system will be given two different options, so that he/she will be able to view the outcomes and choose the most appropriate one for his/her organization. Our system will implement two kinds of algorithms: The first will be the Bayesian algorithm, which operates on the Bayes' statistical formula, and the second one is the admin-controlled percentage-based algorithm, which operates by using the black and white words specified by the administrator. When the first algorithm is activated, the BayesWord table of the database will be used for filtering and when the second algorithm is activated BlackWordList, WhiteWordList, BlackWordGroup, WordInGroup, ExtraWord, and ExtraWordGroup tables will be used for content filtering purposes.

For outgoing packets, the system will be interacting with ConfidentialData table in the database. The words in the outgoing packet will be checked against the keywords provided by the administrator and the ones which include forbidden content over a certain threshold will be blocked.

Information about the blocked packets will be forwarded to the Statistics Module so that the administrator will be informed.

Intended Procedure To Be Followed:

<u>Content Filtering Procedure:</u>

The first algorithm to filter the content coming from remote servers to the local area network will be the Bayesian algorithm, which uses probabilities in order to differentiate the malicious content from others. The algorithm will use the BayesWord table of the database which is filled by the Learning Module. The algorithm will work as follows:

For each word in the incoming packet data;

Retrieve the frequency of this word in malicious packets (a)

Retrieve the frequency of this word in harmless packets (b)

Retrieve the number of malicious packets inspected by Learning Module (c)

Retrieve the number of harmless packets inspected by Learning Module (d)

Calculate;

Malicious Content Probability (MCP) = (a/c) (if MCP is greater than 1.0 set MCP to 1.0)

Harmless Content Probability (HCP) = (b/d) (if HCP is greater than 1.0 set HCP to 1.0)

Maliciousness = MCP / (MCP + HCP)

Select the 20 words with highest Maliciousness and 20 words with lowest Maliciousness to compute the Maliciousness Probability of the packet.

$$\text{Maliciousness Probability} = \frac{\prod_{i=0}^{40} (\text{Maliciousness of Word}_i)}{\prod_{i=0}^{40} (\text{Maliciousness of Word}_i) + \prod_{i=0}^{40} (1 - \text{Maliciousness of Word}_i)}$$

Packet will be blocked if the computed Maliciousness Probability is higher than a predefined threshold.

The second algorithm will give the control of this module to the administrator by letting him/her specify the black and white words through the Web interface. Each black word will be assigned to a constant maliciousness and each white word will be assigned to a lower constant maliciousness. Afterwards, the maliciousness probability will be computed by calculating the expected value of packet's maliciousness i.e.:

$$\text{Maliciousness Probability} = \frac{\sum(\text{Maliciousness of BlackWord}_i * \text{number of occurrences of BlackWord}_i)}{\sum(\text{Maliciousness of Word}_i * \text{number of occurrences of Word}_i)}$$

<u>Confidential Data Hiding Procedure:</u>

Confidential data hiding will be controlled by the administrator through the Web interface. Each keyword indicating a confidential information violation will be assigned to a criticality value (from *crucial* to *negligible*) by the administrator. Afterwards the criticality of the packets will be computes as follows:

$$\text{Criticality} = \sum(\text{Criticality of Keyword}_i * \text{number of occurrences of Keyword}_i)$$

If the criticality is over a certain threshold then the outgoing packet will be blocked.

Input / Output Specifications:

This module retrieves the packets from Network Traffic Monitoring Module and traces them for content. It permits them to pass or it blocks them and sends information about the blocked packets to the Statistics Module for the statistical analysis.

2.4 Restriction Module

Aim of the Module:

Restriction module will block black URLs by defining access rights and limit the overuse of internet by setting download size for user groups and/or users.

General Description and Interactions with the System:

Restriction module of our system will provide two types of restrictions. First type is about URL access restriction and the second one is about download restriction of local clients. URL black list or restricted time interval of URL's will be considered for URL access restriction and local IP's download limit will be the criteria for download restriction.

This module runs in interaction with the database. It obtains the remaining download size of user from the User table and updates the value of the remaining download size.
URL access restriction is also handled by communication with database; module checks the users' black URL's from ExtraURL, ExtraURLGroup, BlackURL and BlackURLGroup tables. From UserInGroup table group of the user should be extracted, from which associating black URL groups can be determined.

Output from this module will trigger the execution of the content filtering module. If the URL is not blocked its content will be inspected by content filtering module.

Intended Procedure To Be Followed:

Restriction module is triggered by users' URL requests. Download restriction control starts the execution by controlling the remaining download size of the user. In case of exceed in the remaining download size of the user, user's request will be blocked and he/she will be warned with a message.

If the user has adequate remaining download size then the control of the restricted URLs will begin. For this purpose firstly WhiteURLList is checked for a match; if there is one then the request will be served. Otherwise both Extra URLs and Extra URL Groups of user and Black URL Group of the user's group should be found. A match among these will result with a block of the access request. In case of blocks, user will be informed.

Input / Output Specifications:

Input will be the source and destination IP's of a URL request. Output will be the page requested or a warning about the restriction.

2.5 Statistics Module

Aim of the Module:

The aim of this module is to provide the necessary feedback to the administrator about the network, so that he/she will have the maximum control on the system.

General Description and Interactions with the System:

The module will be displaying statistics by using the NetworkTrafficLog, ConfidentialDataViolations and FilteredContent tables in the database. The following statistics will be displayed on a daily and monthly basis.

- ***Network Traffic Density:***

  This is the information about the overall network traffic of the organization.

  Daily statistics will be giving information about network traffic density in 2-hour-intervals, such as 9:00-11:00, 11:00-13:00 etc; in units of kilobytes per second. During the day, this page will show the hours up to the current time. Daily network traffic density for each user group will also be displayed individually on administrator's demand.

  Monthly statistics will be giving information about the history of the last month. For the day the administrator wants to see, he/she will be able to specify time intervals and see the statistics associated. Also the busier days of weeks and busier hours of days will be displayed in graphics (traffic density versus time) for easy visualization.

Sample View for Network Traffic Density:

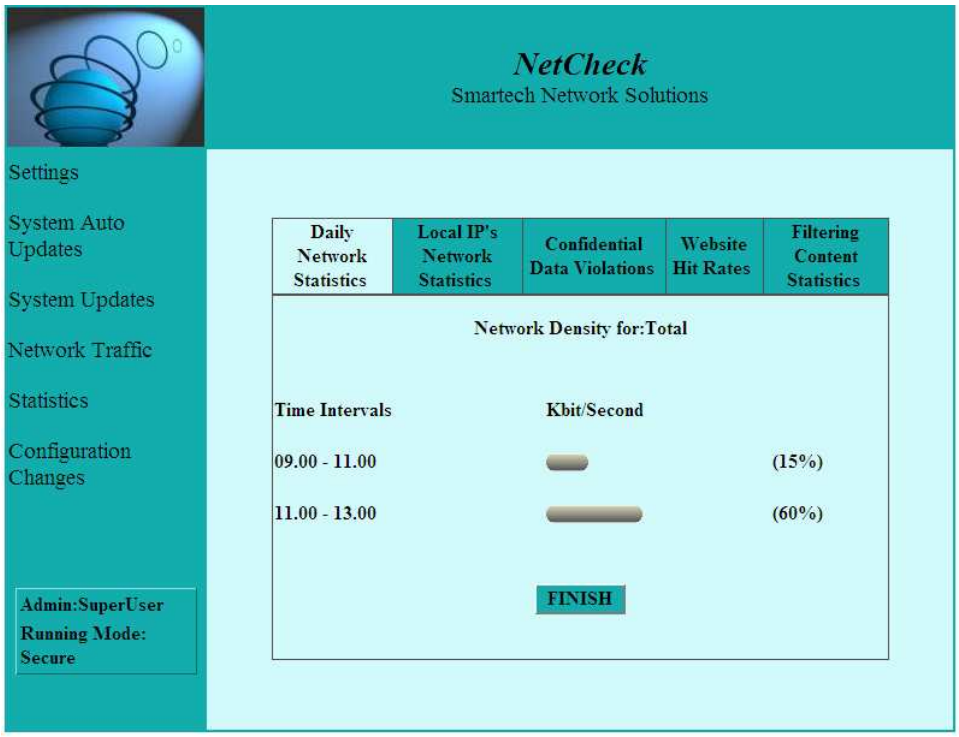


Figure-13: Network Traffic Density Statistics Screen

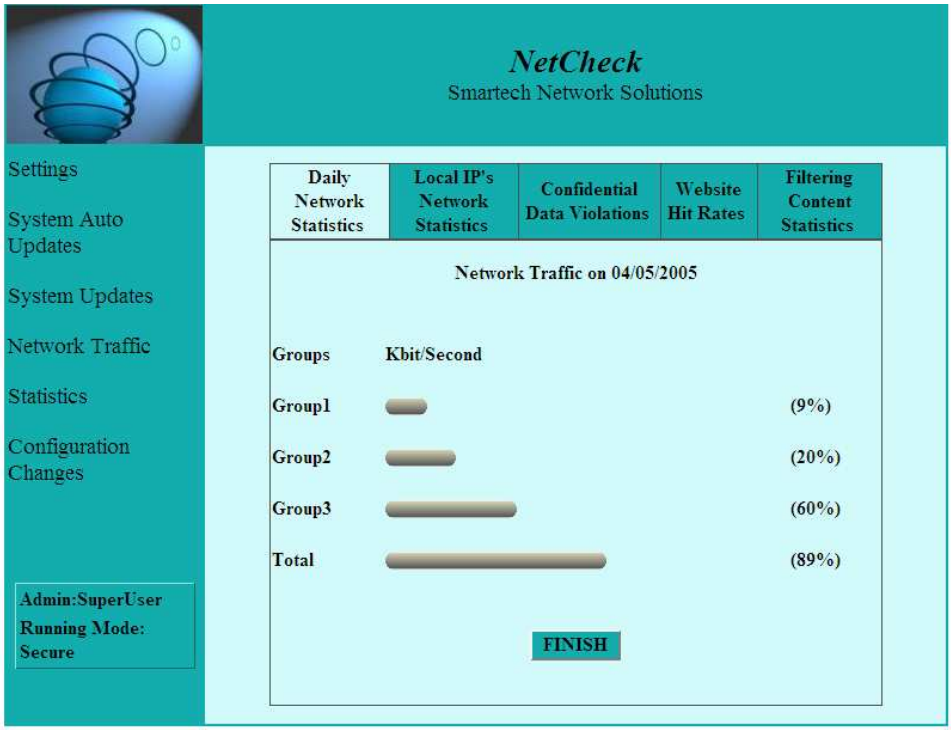


Figure-14: Network Traffic Density Statistics Screen (Time and Group Specified)

- ***Local IP's URL Requests:***

  This is the statistical information about the individual people's URL requests. Administrator will be providing the user's IP and the time interval of concern, and returned information will display the URL's, communication sizes and exact time of the communication.

Sample View for Local IP's URL Requests:

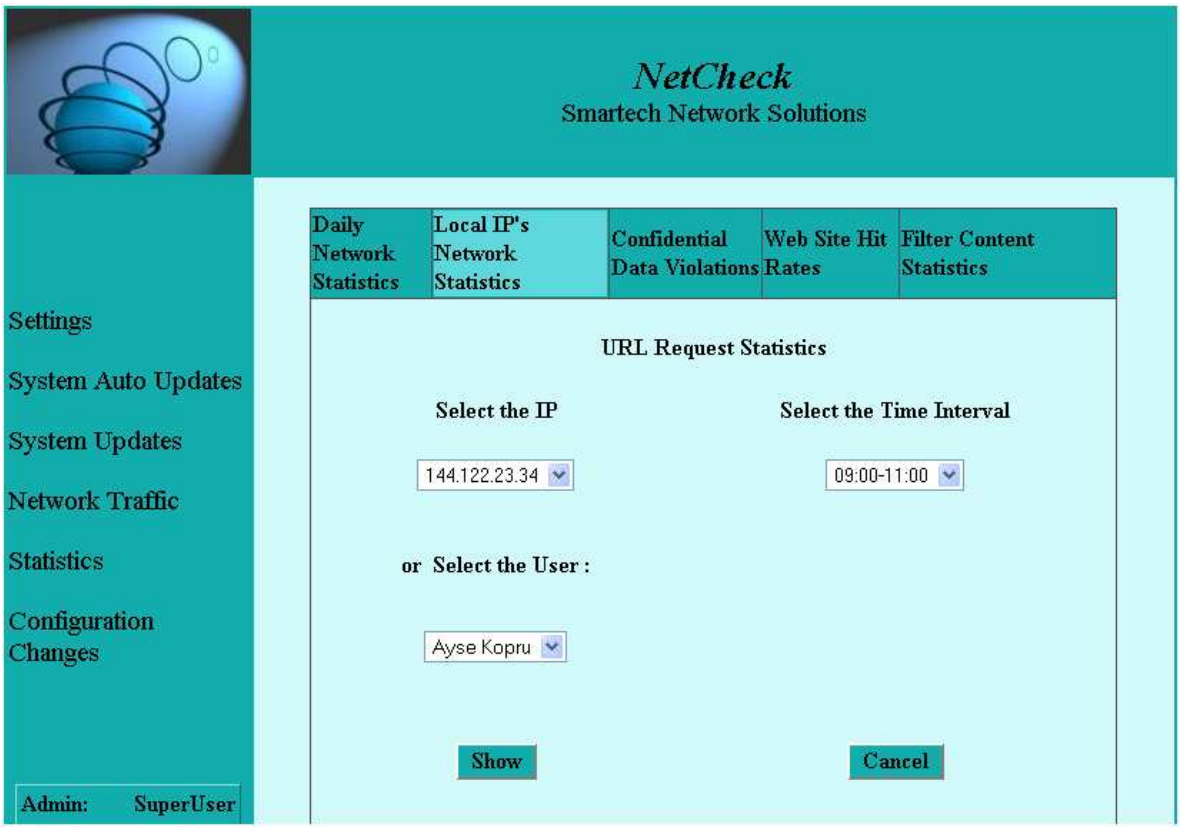


Figure-15: Local IP's URL Request Statistics Screen

- ***Local IP's Download Size:***

  This will be providing the local IP's download size statistics. The administrator will be provided with the information about the user's download limit exceeds.

  Sample View for Local IP's Download Size:

Figure-16: Local IP's Download Information Statistics Screen

- ***Hit Rates of Web Sites:***

  Hit rates of most frequently accessed web sites will be displayed to the system administrators. The administrator will also have the option of viewing web site hit rates of each user group. This statistics will also be displayed in 2-hour-intervals.

Sample View for Hit Rates of Web Sites:

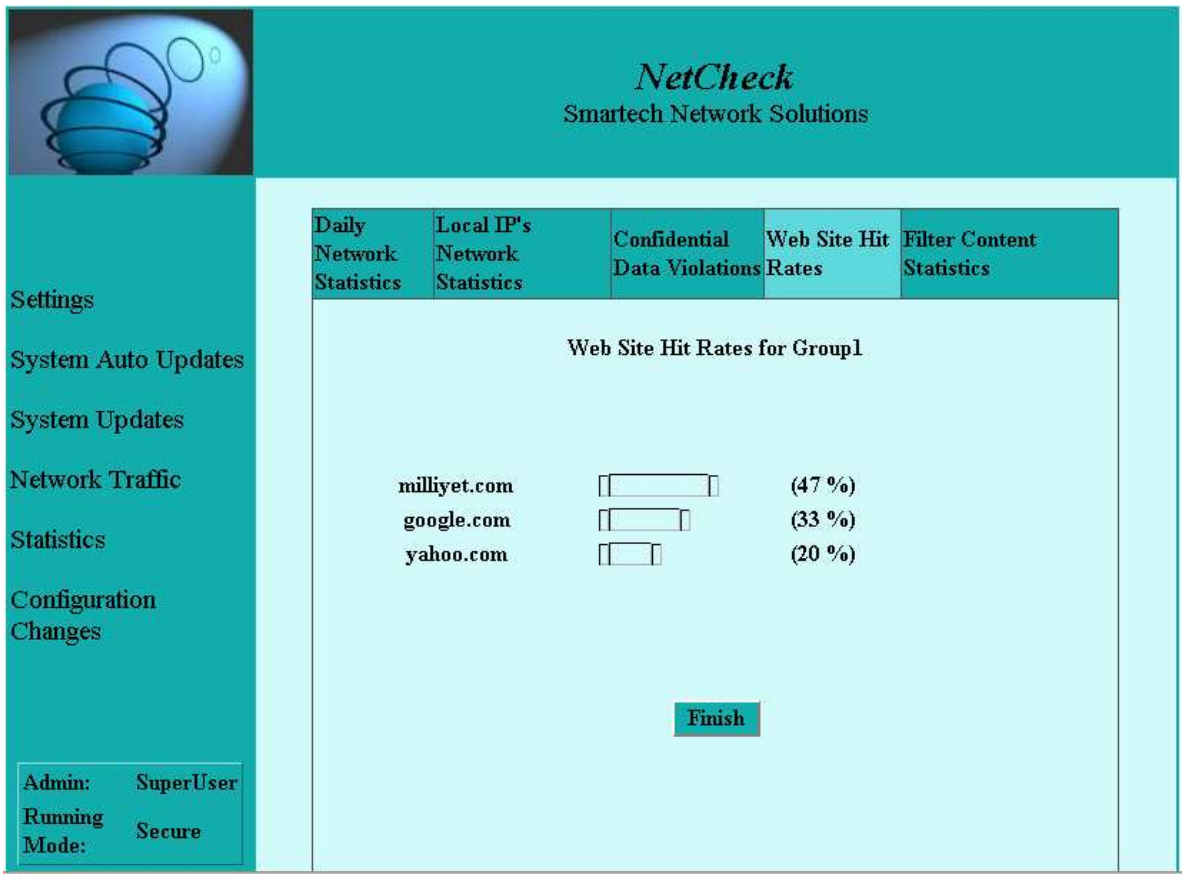


Figure-17: Statistics Related to Hit Rates of Web Sites Screen

- ***Local IP's Violations of Confidential Data Protection:***

  Confidential data violations will be reported to the administrators. A general listing in order of time will be displayed. Also, the administrator will have the option of seeing violations of a certain IP and/or in a certain time interval. Information about the IP, violated confidential data and exact time will be shown. The displayed data will be the most significant five violations.

Sample View for Local IP's Violations of Confidential Data Violations:

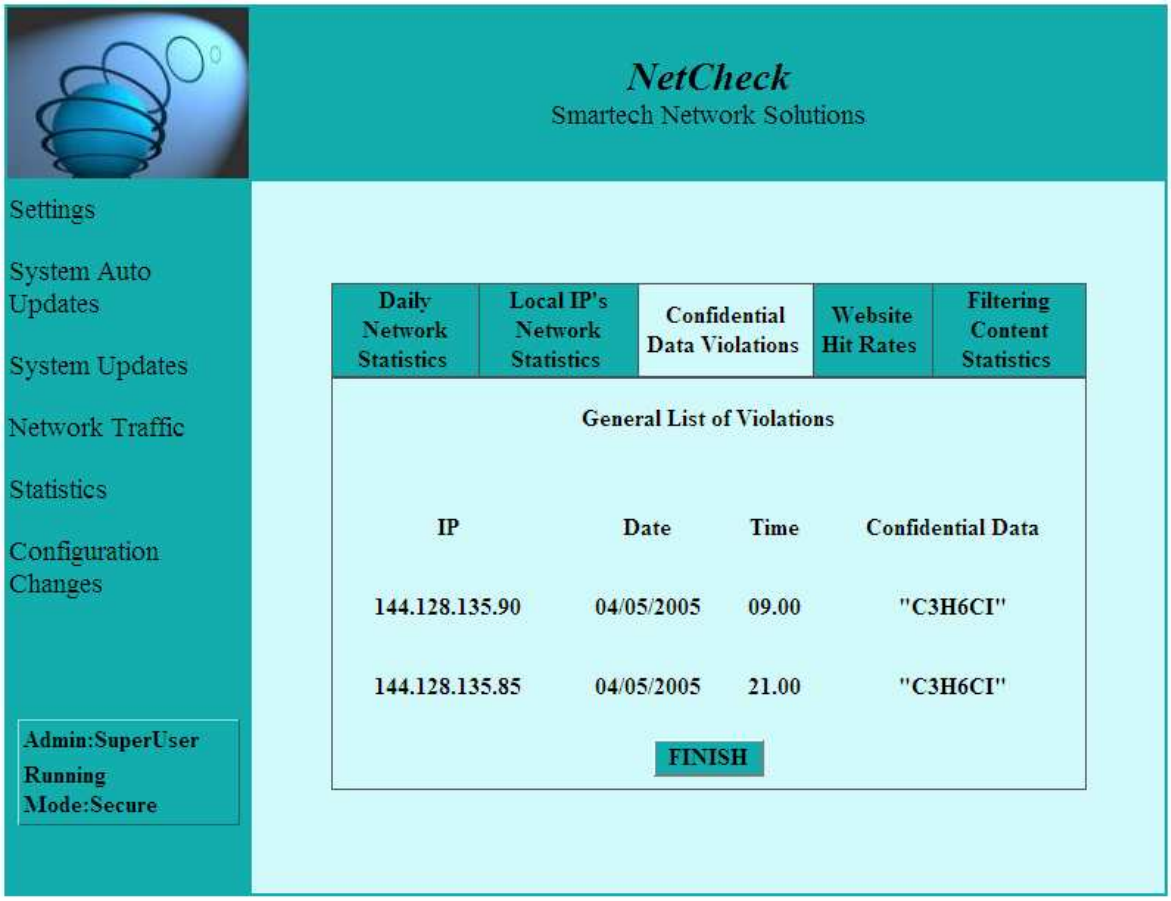


Figure-18: Statistics Related to Confidential Data Violations Screen

- ***Filtered Content:***

  The administrator will be able to view the filtering actions which have been carried out according to the content filtering settings he/she specified. So the administrator will have the option of viewing the results of current settings and change them if necessary. The information displayed will be the filtered URL, sourceIP, destinationIP, time and violated settings(i.e., the most significant keywords that caused the filtering action). The administrator will have the option of seeing filtered packets requested by a certain IP and/or in a certain time interval.

Sample View for Filtered Content:

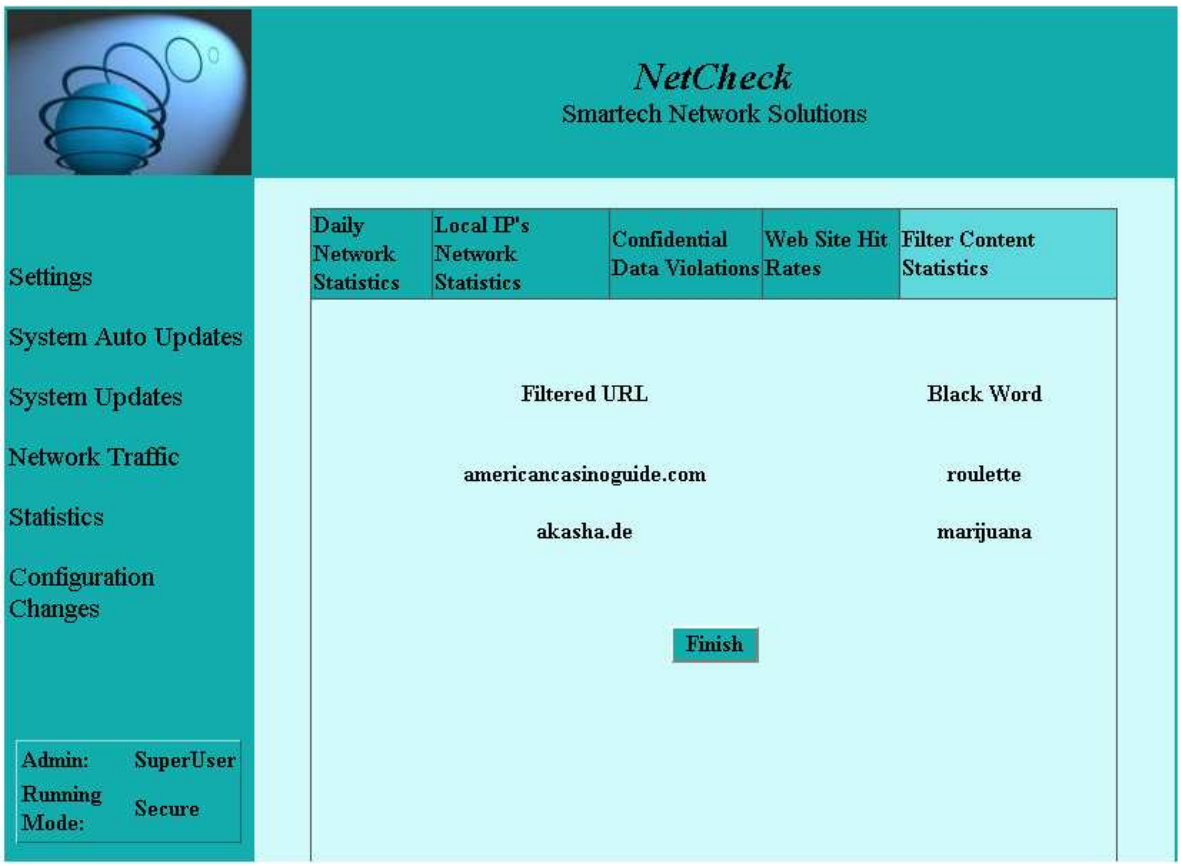


Figure-19: Statistics Related to Content Filtering Screen

Intended Procedure To Be Followed:

This module will be activated when the administrator sends a request for seeing the statistical data, via the web interface. The module will then retrieve the associated data from the database and do the necessary calculations.

Input / Output Specifications:

The input of the module is the request of the administrator, and the entries from the NetworkTrafficLog, ConfidentialDataViolations and FilteredContent tables. The output of the module will be the computed statistics that will be displayed on the web interface.

2.6 Logging Module

Aim of the Module:

Logging module will create a log file listing recent actions of the administrator in a human readable format and that file will be composed of lines in the following format:

user name of the administrator | performed action | configured table's name|
old configuration | new configuration

By enabling a feature like this we provide the administrator with an undo option for recently applied updates to the system so that administrator may return back to the old settings.

General Description and Interactions with the System:

System will have a trigger for main database table updates that are associated with general setting modifications applied by the administrator. The updates to the Administrator, WhiteWordList, BlackWordList, WhiteURLList, BlackURLList, BlackWordGroup, BlackURLGroup and ConfidentialData tables will be saved in the configuration log file.

Database updates are enforced via insert, delete or update SQL statements. For handling undo and redo operations efficiently, logging module stores the corresponding SQL statements of old and new configurations. This feature requires handling of each SQL statement type differently; i.e., for an insert operation, old configuration holds a delete SQL statement, for a delete operation it holds an insert statement and for updates it has another update as its former setting.

Action performed is also constructed depending on the SQL statement type; for insert statements, performed action holds the primary key of the added tuple, for delete statements, primary key of the deleted row will be written to the configuration log file, whereas update statements will have a matching action that holds primary key of the modified row, and old and new values of the changed column.

For example casino being inserted to BlackWordList table should be shown in the configuration log file as:

SuperUser | “insertion casino” | BlackWordList
| ”delete from BlackWordList where word=’casino’”
|”insert into BlackWordList values(‘casino’)”

Intended Procedure To Be Followed:

In case of an update on a database table, logging module will be triggered. Before applying modifications to the intended tables, mechanism for saving old configuration will run. Then the update will be performed. Afterwards new configuration field of the log file will be set with the corresponding SQL statement. Finally the action performed and database table fields of the configuration log file will be set.

Input / Output Specifications:

Input to the module is an SQL statement which applies an update to the database tables of the system. Output will be a line in the configuration log file showing the enforced update.

Sample View of the Module:

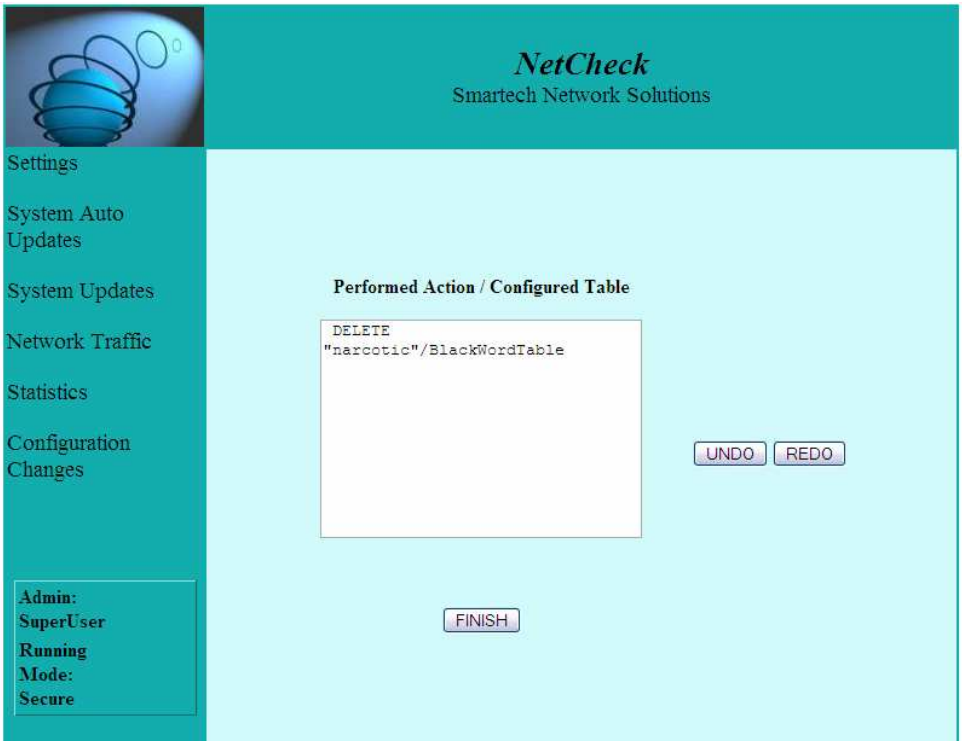


Figure-20: Viewing Configuration Logs Screen

2.7 Learning Module

Aim of the Module:

This module calculates the malicious occurrences and harmless occurrences of words for Bayesian content filtering algorithm.

General Description and Interactions with the System:

This module will be activated by the administrator via web interface and it will be running independently from the rest of the system. This module will be filling the BayesWord table in the database which consists of the word, malicious occurrences of the word, and harmless occurrences of the word.

Intended Procedure To Be Followed:

The administrator will specify a URL and indicate whether it is a malicious or harmless one. The system will then fetch packets from the URL. If the URL is a malicious one, the malicious occurrences attribute of this word will be incremented for each occurrence of the word. Otherwise, the harmless occurrence attribute will be incremented.

Input / Output Specifications:

This module will only be updating the BayesWord table in the database. It will be using the contents of URLs which are specified by the administrator.

2.8 Modifications in the System Modules

We have mentioned in our initial design report that our system will make use of a caching mechanism; in order to make access to the most frequently requested sites faster. However, having elaborated our design, we have come to a decision that caching indeed comes with more disadvantages than advantages. We will not integrate an existing proxy to our system, which means that the system would have to handle this task, in addition to its main tasks. One of the main goals of our system is outstanding performance, which was the reason that we have thought of a cache mechanism in the first place. If the caching mechanism is included, the system must decide on the most frequently accessed URLs, track their packets and group them in caching storage, and compare each request with this cached packets' information. If the packets exist, the system must then act as if it were a web server, forming the response and sending it back to the client. In addition to these concerns, we have noted that pages which contain PHP forms must not be cached. What is more, any cached URL must be timed-out in a rather short time interval (for instance, 15 minutes) so that continuously changing sites (i.e. www.milliyet.com) will not be displayed the same way all day long.

Considering all these issues, we have concluded that caching mechanism will slow the overall system for no outstanding profits. So we have decided to exclude this feature from our product.

The next issue that we have considered in our detailed design is the auto-updates of the system. We have decided to add an auto-update mechanism considering that the black URLs are increasing continuously. System will be uploading the BlackURLList and WhiteURLList tables in the database by fetching BlackURL and WhiteURL suppliers' web sites (including the product's web site) contents. This procedure will be activated in certain intervals.

For the further releases of our product, we are planning to implement two more extensions. The first extension is the categorization of visited web sites according to the outputs of Bayesian algorithm. For this, we are planning to divide the BayesWords into topic categories. The second one is about hiding the internal data of the computers in the local area network such as the operating system, web server, database server etc.

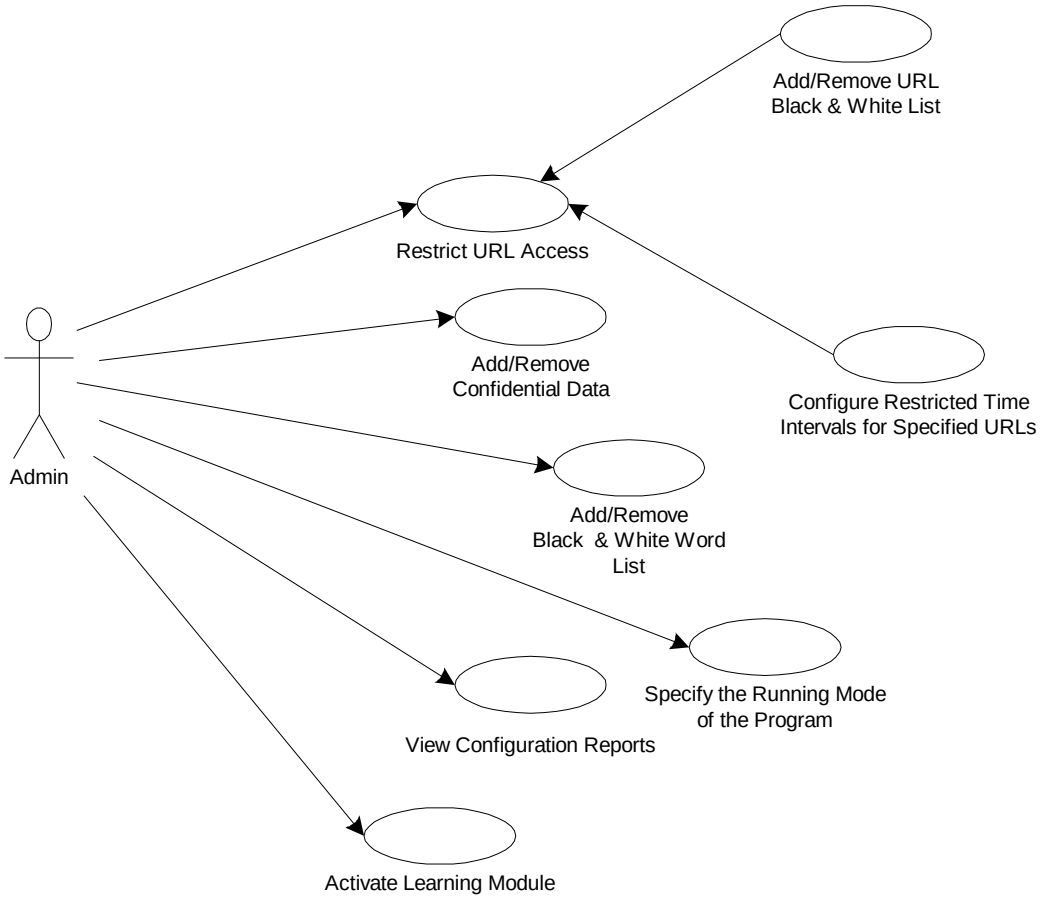
Auto-Update Interface:

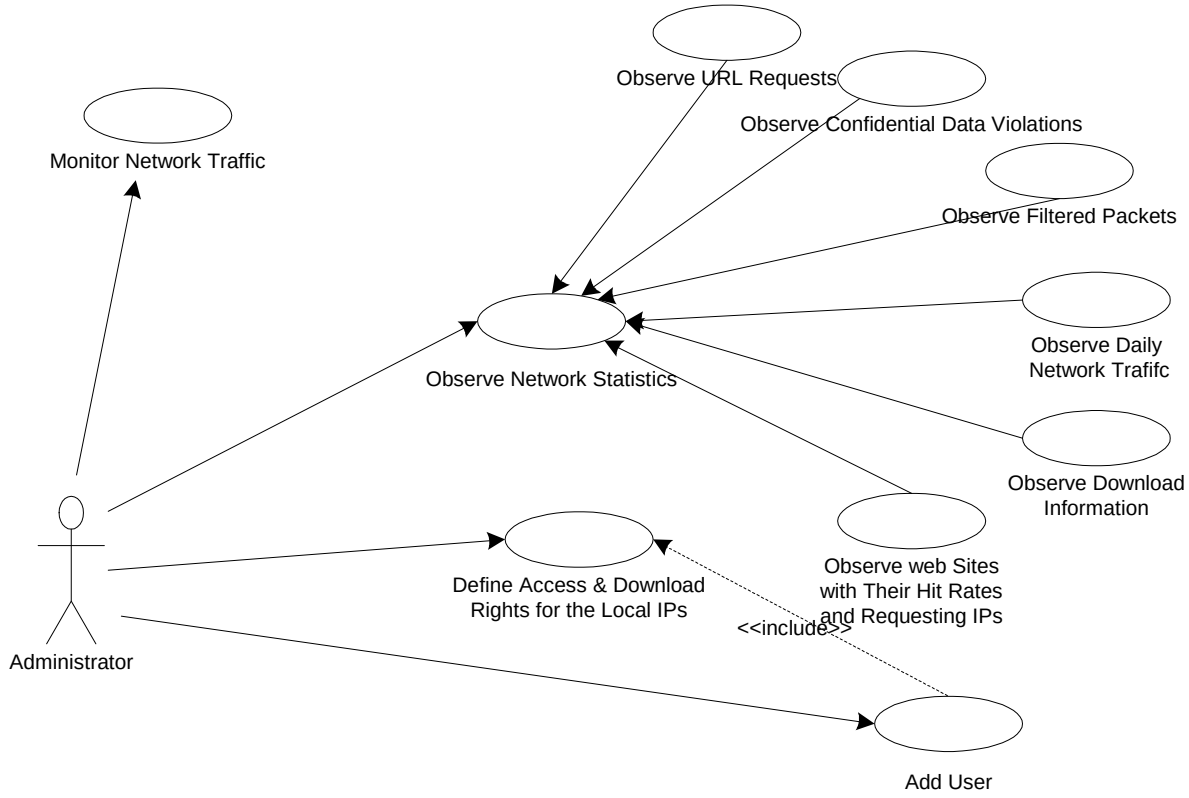
Figure-21: Update Activation Screen

3 SYSTEM DESIGN

3.1 Use Case Diagrams

3.1.1 Use Case for Administrator





3.1.2 Scenarios for Administrator Use Case

The administrator of the system will be controlling our product via a web interface. Through the interface, the administrator will have the following controls over the system:

Monitor Network Traffic

Basic Flow of Control: Administrator will have the option of viewing network traffic in real time. Source IP, destination IP, accessed URL, packet size and time information of each packet will be displayed via the web interface.

Restrict URL Access

Basic Flow of Control: Administrator may *Add/Remove URL Black and White List*, *Configure Restricted Time Intervals for Specified URLs*.

Add/Remove Confidential Data

Basic Flow of Control: Administrator can specify certain keywords that will not be allowed to be included in outgoing packets. The administrator must also specify the criticality of each keyword.

Add/Remove URL Black and White List

Basic Flow of Control: Administrator can configure URL Black List Table, which contains the URLs that are forbidden to be accessed by local clients, or the URL White List Table, which contains URLs that will not be blocked in any case.

Configure Restricted Time Intervals for Specified URLs

Basic Flow of Control: Administrator may specify time limitations for accessing some sites. For instance, newspaper sites may be forbidden during the morning, when network traffic is especially busy.

Define Access and Download Rights for Local IPs

Basic Flow of Control: Download and access rights can be granted to user groups and/or individual users. Download rights indicate the maximum packet size limit that can be downloaded in a day. Access rights define which entries of black URL / word list apply to the user / user group.

Add/Remove Black and White Word List

Basic Flow of Control: Administrator can configure Black Word List Table, which contains the word that will be used in content filtering, or the White Word List Table, which contains words that will not be filtered in any case.

Observe Network Statistics

Basic Flow of Control: Administrator may *Observe Web Sites with Their Hit Rates and Requesting IPs*, *Observe Download Information*, *Observe Daily Network Traffic*, *Observe Filtered Packets*, *Observe Confidential Data Violations*, and *Observe URL Requests.*

View Configuration Reports

Basic Flow of Control: Administrator can view the reports about latest configurations made to the system.

Activate Learning Module

Basic Flow of Control: The administrator may activate Learning Module from time to time if he/she has been selected Bayesian algorithm for content filtering.

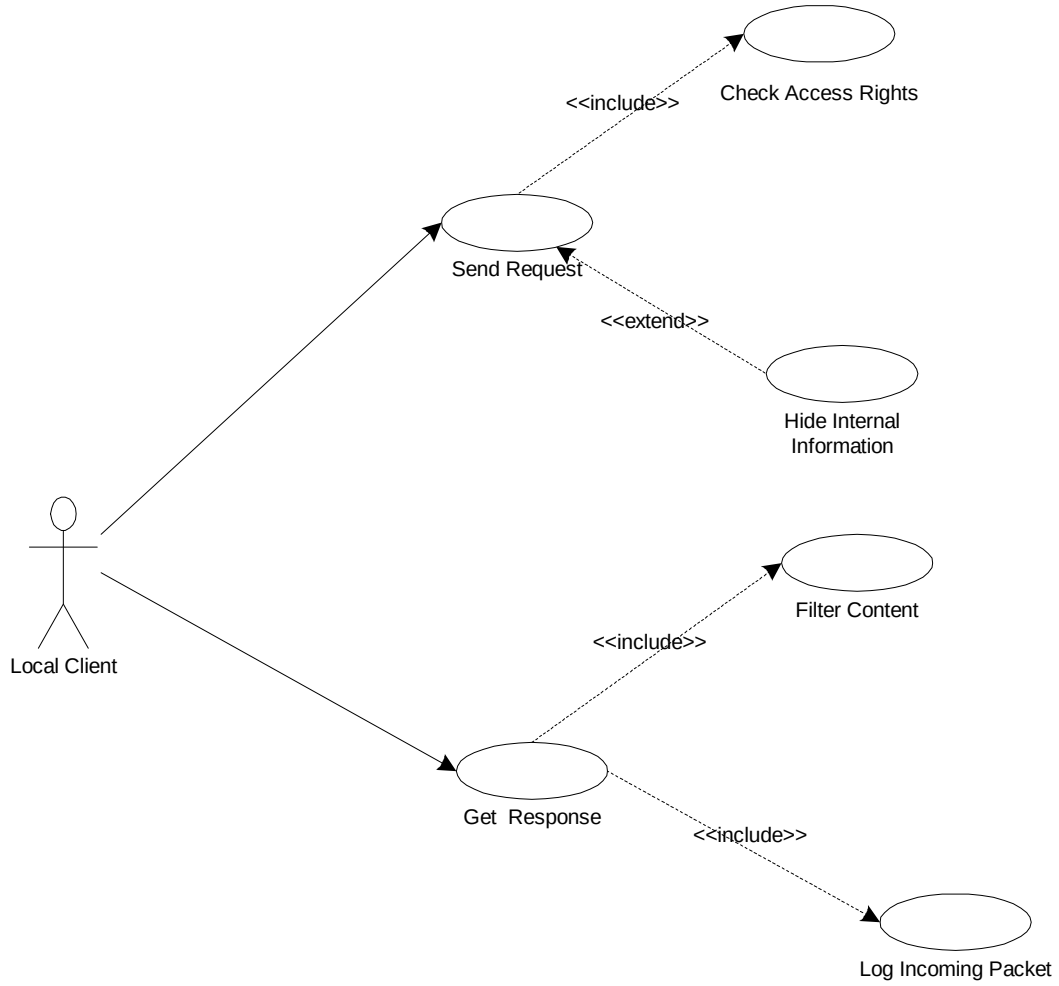
Add User

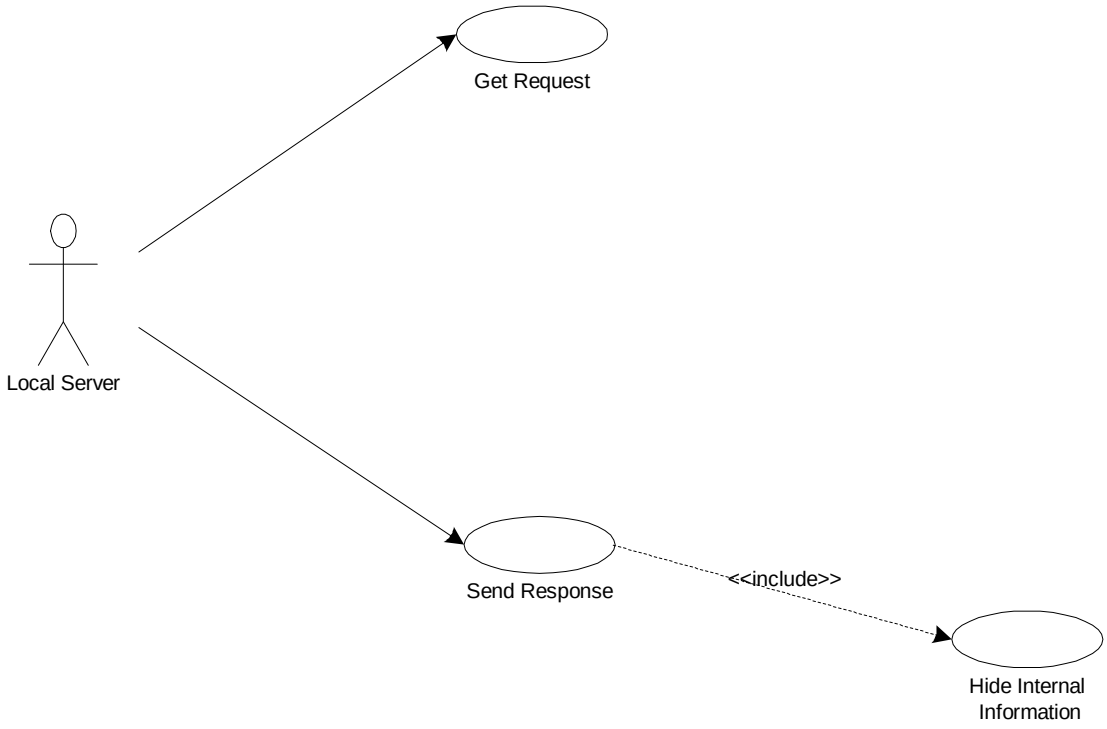
Basic Flow of Control: New users are added via web interface. *Include (Define Access and Download Rights for Local IPs).*

Specify the Running Mode

Basic Flow of Control: Administrator may choose between different modes of the program, which offer different functionalities.

3.1.3 Use Case for Local Client and Server





3.1.4 Scenarios for Local Client and Server Use Case

Send Request

Basic Flow of Control: Local client sends a request then *include (Check Access Rights).* If the request does not violate the access rights then *Hide Internal Information.*

Alternative Flow of Control: If the local client's request fail to satisfy *include (Check Access Rights)* then an error message will be displayed to the client.

Check Access Rights

Basic Flow of Control: Request packet will be first inspected to control if the destination address is in the white list. If the destination IP is in the white list, request will be served without any further considerations. However in case when the destination address does not exist in the white list, source address will be taken into account to check for restrictions on the black list table. If no violations are detected permission will be granted to the user.

Hide Internal Information

Basic Flow of Control: Company oriented security policies will be detected in order to prevent the private data from being sent out and NAT (Network Address Translation) will be implemented to local client IPs.

Get Response

Basic Flow of Control: Local client gets the response from the remote server. *Include (Filter Content) and include (Log Incoming Packet).*

Alternative Flow of Control: If the incoming packet fail to satisfy *include (Filter Content)* then an error message will be displayed to the client

Filter Content

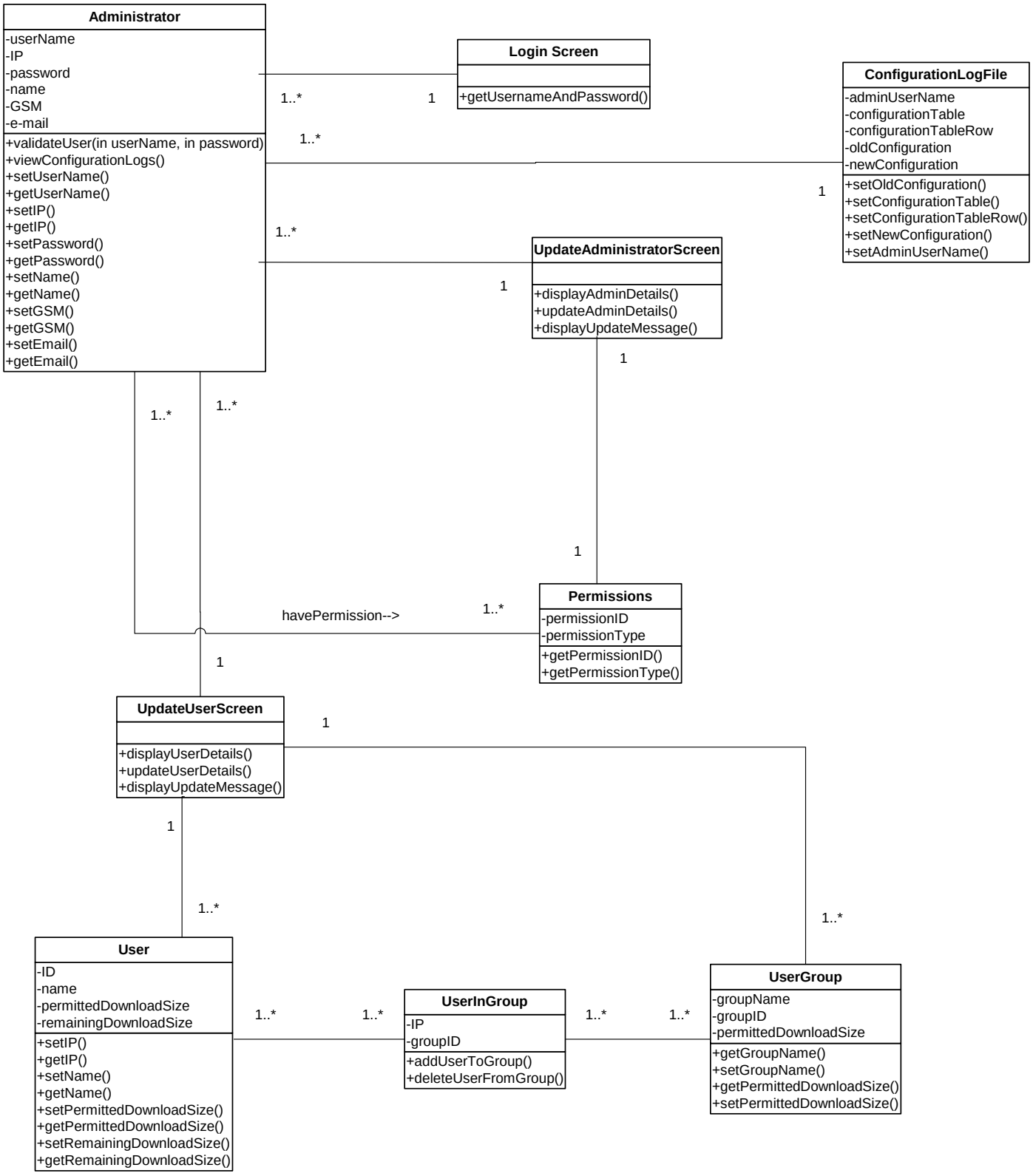
Basic Flow of Control: Incoming packet will be first inspected to control if it violates the selected content filtering algorithm. If the packet does not violate the selected algorithm, the packet will be allowed to pass.

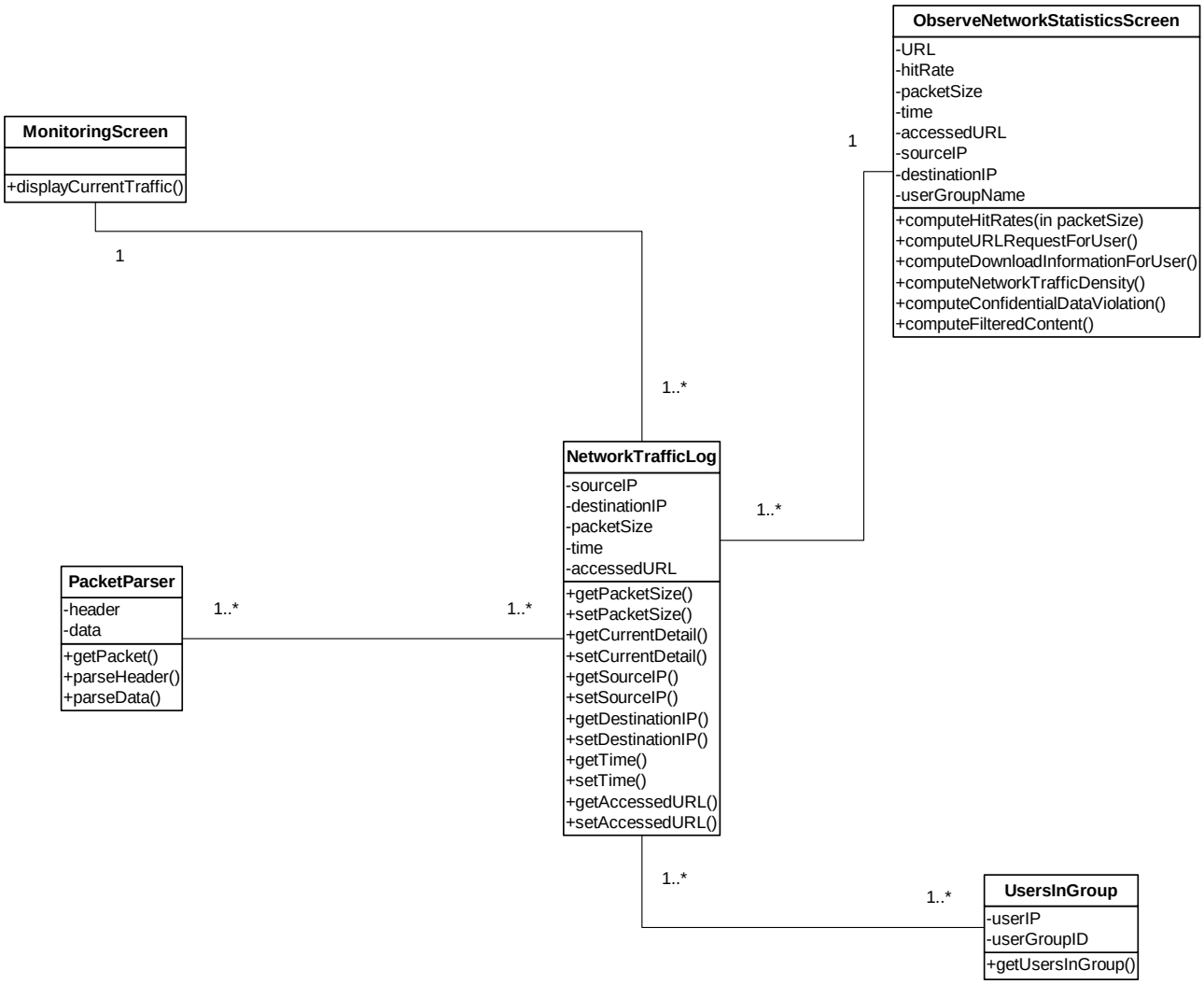
Alternative Flow of Control: If the packet violates the selected algorithm, the packet will not be allowed. An error message will be displayed to the client.

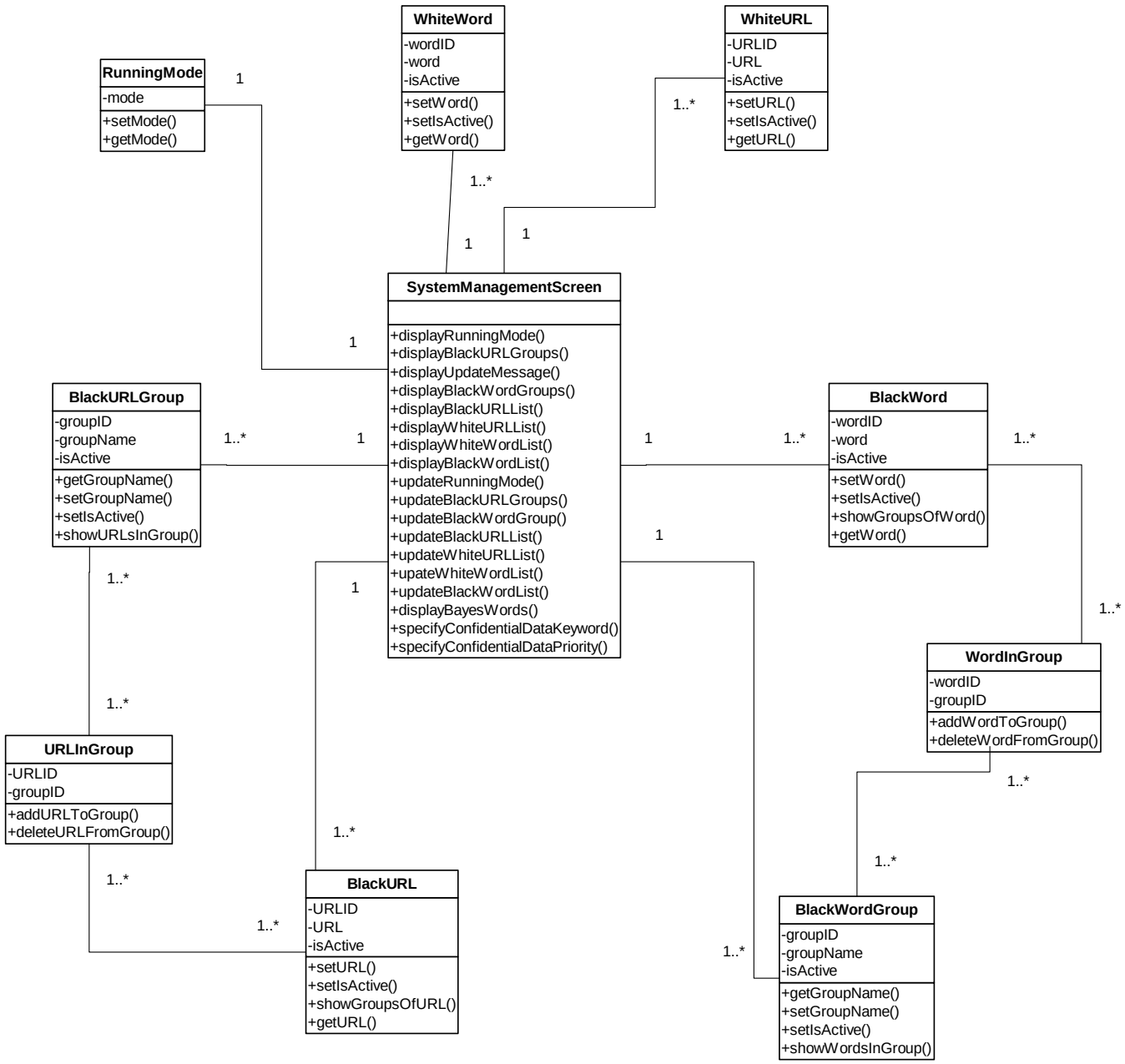
Log Incoming Packet

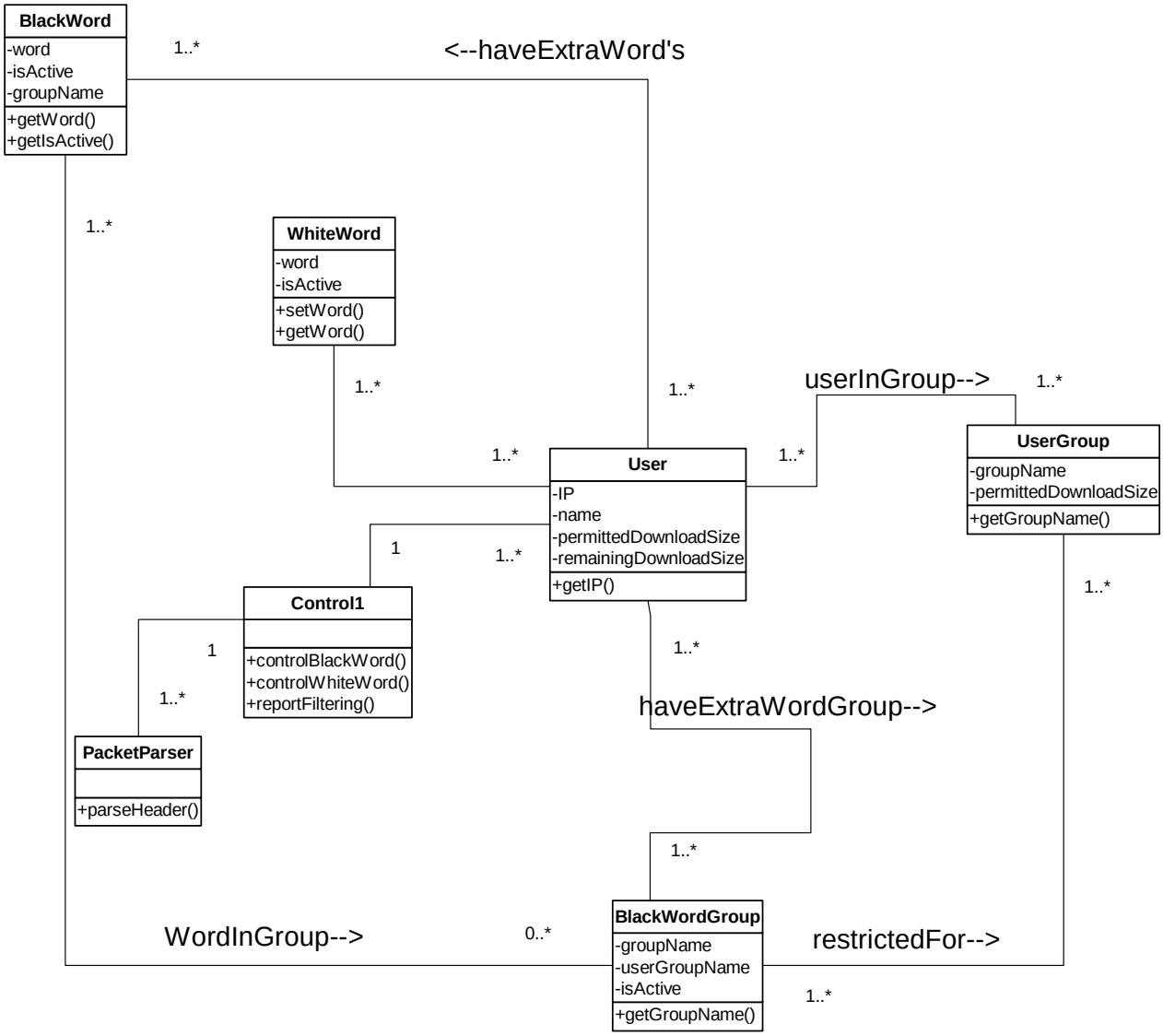
Basic Flow of Control: Content of the incoming packet is logged for further usage.

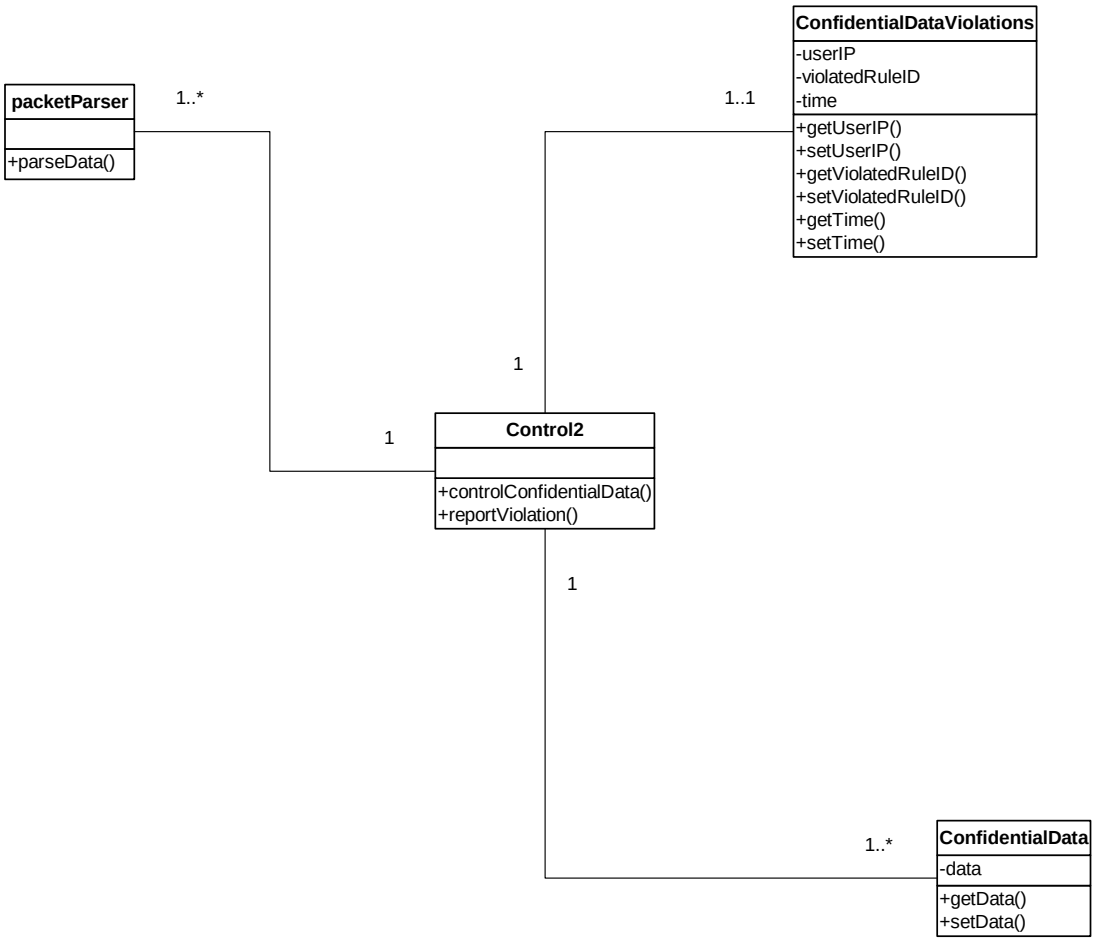
3.2 Class Diagrams

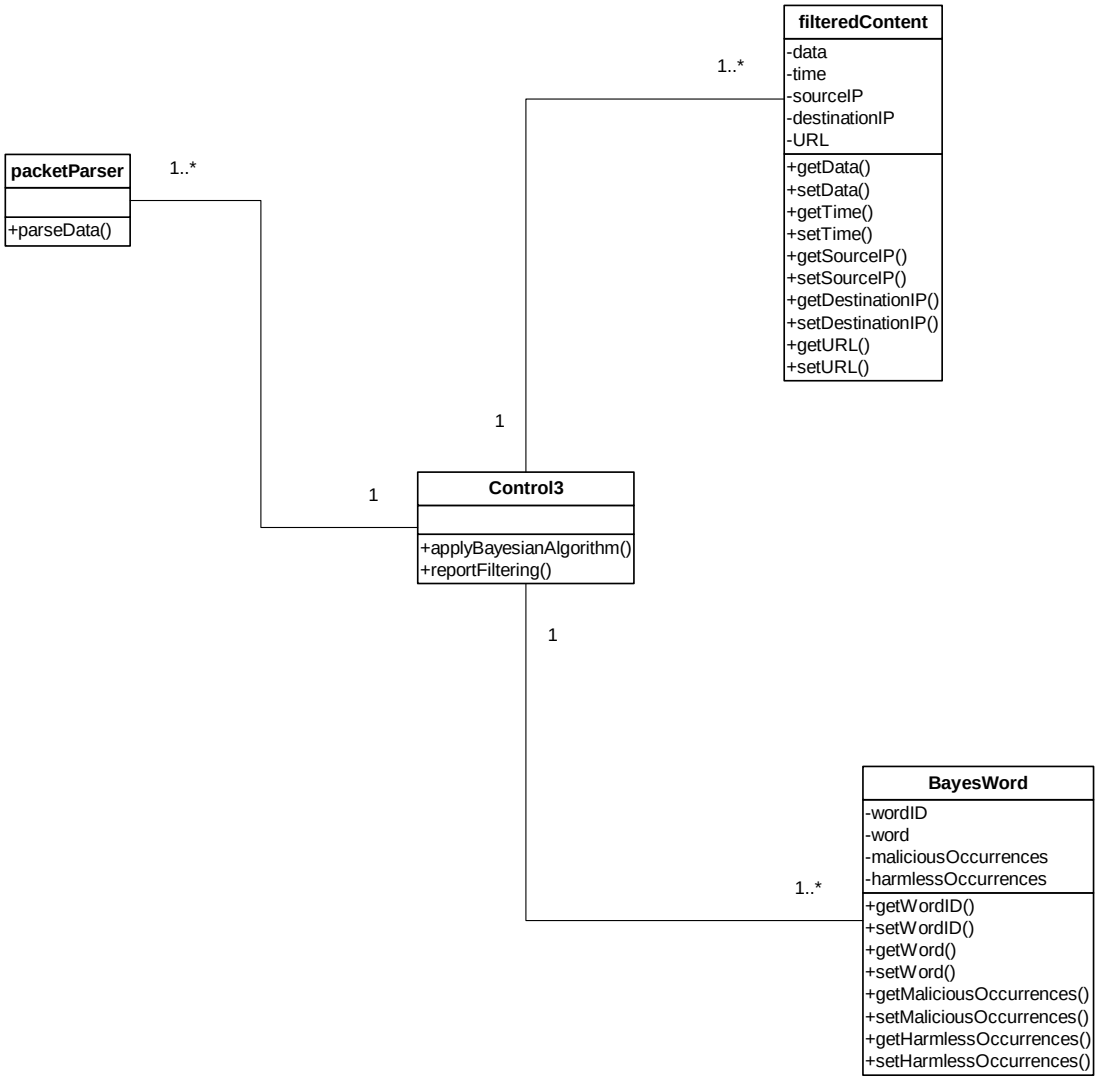


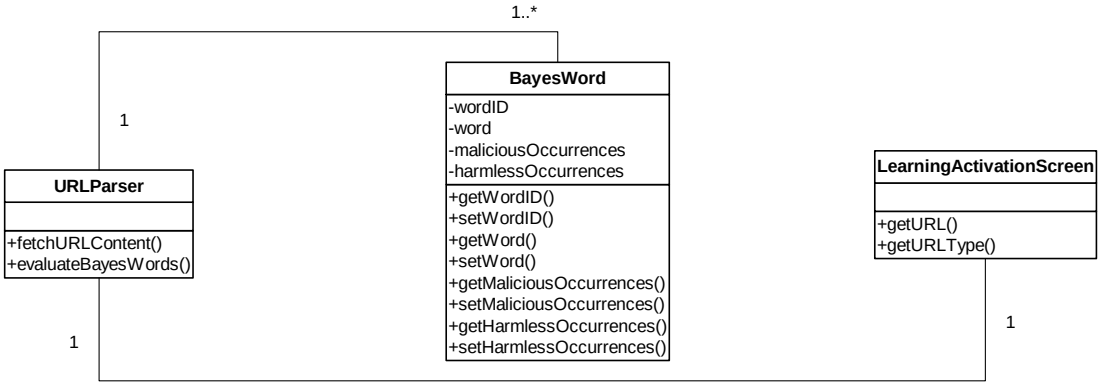

ObserveNetworkStatisticsScreen
-URL
-hitRate
-packetSize
-time
-accessedURL
-sourceIP
-destinationIP
-userGroupName
+computeHitRates(in packetSize)
+computeURLRequestForUser()
+computeDownloadInformationForUser()
+computeNetworkTrafficDensity()
+computeConfidentialDataViolation()
+computeFilteredContent()
MonitoringScreen
+displayCurrentTraffic()
1
1
1..*
NetworkTrafficLog
-sourceIP
-destinationIP
-packetSize
-time
-accessedURL
+getPacketSize()
+setPacketSize()
+getCurrentDetail()
+setCurrentDetail()
+getSourceIP()
+setSourceIP()
+getDestinationIP()
+setDestinationIP()
+getTime()
+setTime()
+getAccessedURL()
+setAccessedURL()
1..*
PacketParser
-header
-data
+getPacket()
+parseHeader()
+parseData()
1..*
1..*
1..*
1..*
UsersInGroup
-userIP
-userGroupID
+getUsersInGroup()


RunningMode
-mode
+setMode()
+getMode()
WhiteWord
-wordID
-word
-isActive
+setWord()
+setIsActive()
+getWord()
WhiteURL
-URLID
-URL
-isActive
+setURL()
+setIsActive()
+getURL()
SystemManagementScreen
+displayRunningMode()
+displayBlackURLGroups()
+displayUpdateMessage()
+displayBlackWordGroups()
+displayBlackURLList()
+displayWhiteURLList()
+displayWhiteWordList()
+displayBlackWordList()
+updateRunningMode()
+updateBlackURLGroups()
+updateBlackWordGroup()
+updateBlackURLList()
+updateWhiteURLList()
+upateWhiteWordList()
+updateBlackWordList()
+displayBayesWords()
+specifyConfidentialDataKeyword()
+specifyConfidentialDataPriority()
BlackURLGroup
-groupID
-groupName
-isActive
+getGroupName()
+setGroupName()
+setIsActive()
+showURLsInGroup()
BlackWord
-wordID
-word
-isActive
+setWord()
+setIsActive()
+showGroupsOfWord()
+getWord()
WordInGroup
-wordID
-groupID
+addWordToGroup()
+deleteWordFromGroup()
URLInGroup
-URLID
-groupID
+addURLToGroup()
+deleteURLFromGroup()
BlackURL
-URLID
-URL
-isActive
+setURL()
+setIsActive()
+showGroupsOfURL()
+getURL()
BlackWordGroup
-groupID
-groupName
-isActive
+getGroupName()
+setGroupName()
+setIsActive()
+showWordsInGroup()
1
1..*


BlackWord
-word
-isActive
-groupName
+getWord()
+getIsActive()
1..*
<--haveExtraWord's
1..*
WhiteWord
-word
-isActive
+setWord()
+getWord()
1..*
1..*
userInGroup-->
1..*
1..*
User
-IP
-name
-permittedDownloadSize
-remainingDownloadSize
+getIP()
1..*
UserGroup
-groupName
-permittedDownloadSize
+getGroupName()
1
1..*
Control1
+controlBlackWord()
+controlWhiteWord()
+reportFiltering()
1
1..*
1..*
haveExtraWordGroup-->
1..*
PacketParser
+parseHeader()
1..*
WordInGroup-->
0..*
BlackWordGroup
-groupName
-userGroupName
-isActive
+getGroupName()
restrictedFor-->
1..*


packetParser
+parseData()
1..*
1
ConfidentialDataViolations
-userIP
-violatedRuleID
-time
+getUserIP()
+setUserIP()
+getViolatedRuleID()
+setViolatedRuleID()
+getTime()
+setTime()
1..1
1
Control2
+controlConfidentialData()
+reportViolation()
1
1..*
ConfidentialData
-data
+getData()
+setData()


filteredContent
-data
-time
-sourceIP
-destinationIP
-URL
+getData()
+setData()
+getTime()
+setTime()
+getSourceIP()
+setSourceIP()
+getDestinationIP()
+setDestinationIP()
+getURL()
+setURL()
1..*
packetParser
+parseData()
1..*
1
1
Control3
+applyBayesianAlgorithm()
+reportFiltering()
1
BayesWord
-wordID
-word
-maliciousOccurrences
-harmlessOccurrences
+getWordID()
+setWordID()
+getWord()
+setWord()
+getMaliciousOccurrences()
+setMaliciousOccurrences()
+getHarmlessOccurrences()
+setHarmlessOccurrences()
1..*


1..*
1
BayesWord
-wordID
-word
-maliciousOccurrences
-harmlessOccurrences
+getWordID()
+setWordID()
+getWord()
+setWord()
+getMaliciousOccurrences()
+setMaliciousOccurrences()
+getHarmlessOccurrences()
+setHarmlessOccurrences()
URLParser
+fetchURLContent()
+evaluateBayesWords()
LearningActivationScreen
+getURL()
+getURLType()
1
1

3.2.1 Class Descriptions

This section is dedicated to the descriptions of the classes shown schematically above. The attributes and the methods of the classes are explained in detail.

LoginScreen Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
getUserNameAnd Password	username: *string* password: *string*	Void	Gets the username and password of the administrators via web interface for authentication purposes

UpdateAdministratorScreen Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
displayAdminDetails	userName: *string*	Administrator	Displays the details of a specified administrator on the web interface
updateAdminDetails	userName: *string*	Boolean	Updates the administrator's details (if applicable)
displayUpdateMessage	boolean	String	Displays a success or failure message on the screen

Administrator Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
userName	string	The user name that the administrator will use to log in to the system
password	string	The password of the administrator
IP	char(15)	The IP of the administrator's computer
Name	string	Administrator's name and surname
GSM	string	The GSM number of the administrator
Email	string	The e-mail address of the administrator

Methods of the Class:

Method Name	Parameters	Return Type	Description
validateUser	username: *string* password: *string*	Boolean	Checks the authentication and authorization of an administrator when he/she logs in
viewConfiguration Logs	void	Configuration LogFile	Fetches the log file records that list the changes made by this administrator to the system settings
getUserName	void	String	Retrieves userName attribute
setUserName	username: *string*	Void	Assigns userName attribute
getIP	void	char(15)	Retrieves IP attribute
setIP	IP: *char(15)*	Void	Assigns IP attribute
getPassword	void	String	Retrieves password attribute
setPassword	password: *string*	Void	Assigns password attribute
getName	void	String	Retrieves name attribute
setName	name: *string*	Void	Assigns name attribute
getGSM	void	String	Retrieves GSM attribute
setGSM	GSM: *string*	Void	Assigns GSM attribute
getEmail	void	String	Retrieves email attribute
setEmail	email: *string*	Void	Assigns email attribute

ConfigurationLogFile Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
adminUserName	string	The user name of the administrator who has changed the system settings
configurationTable	string	The table that has been changed
configurationTableRow	integer	The row of the table that has been changed (applicable only if the change is an insertion or update)
oldConfiguration	string	Description of the previous configuration
newConfiguration	string	Description of the new configuration

Methods of the Class:

Method Name	Parameters	Return Type	Description
getOld Configuration	Void	String	Retrieves oldConfiguration attribute
setOld Configuration	oldConfiguration: *string*	Void	Assigns oldConfiguration attribute
getConfiguration Table	Void	String	Retrieves configurationTable attribute
setConfiguration Table	configuration Table: *string*	Void	Assigns configurationTable attribute
getConfiguration TableRow	Void	Integer	Retrieves configurationTableRow attribute
setConfiguration TableRow	configuration TableRow: *integer*	Void	Assigns configurationTableRow attribute
getNew Configuration	void	String	Retrieves newConfiguration attribute
setNew Configuration	newConfiguration : *string*	Void	Assigns newConfiguration attribute
getAdmin UserName	void	String	Retrieves adminUserName attribute
setAdmin UserName	adminUser Name: *string*	Void	Assigns adminUserName attribute

Permissions Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
permissionID	integer	A unique integer to specify the permissions that administrators have
permissionType	string	Type of the permission which specifies the administrators' degree of control over the system

Methods of the Class:

Method Name	Parameters	Return Type	Description
getPermissionID	void	Integer	Retrieves permissionID attribute of the permissions class
getPermissionType	void	String	Retrieves permissionType attribute of the permissions class

User Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
IP	char(15)	The IP of the user
Name	string	The name of the user
permittedDownloadSize	float	The extent which the user is limited to access Internet
remainingDownloadSize	float	The user's download size lowered to the remaining download size after each connection to Internet

Methods of the Class:

Method Name	Parameters	Return Type	Description
getIP	void	char(15)	Retrieves the IP of the user
setIP	IP: *char(15)*	Void	Assigns the IP of the user
getName	void	String	Retrieves the name of the user
setName	name: *string*	Void	Assigns the name of the user
getPermittedDownload Size	void	Float	Retrieves the permitted download size of the user
setPermittedDownload Size	permitted DownloadSize: *float*	Void	Initially assigns download size to the user
getRemainingDownload Size	void	Float	Retrieves the remaining download size of the user
setRemainingDownload Size	remaining DownloadSize: *float*	Void	Assigns remaining download size to the user after each connection to the Internet

UpdateUserScreen Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
displayUserDetails	IP: *char(15)*	User	Displays the details of a specified user on the web interface
updateUserDetails	IP: *char(15)*	Boolean	Updates the user details (if applicable)
displayUpdate Message	boolean	String	Displays a success or failure message on the screen

UserGroup Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
groupName	string	The name describing the user group
groupID	string	The ID of the group
permittedDownloadSize	integer	The default download size that will limit the members of the group

Methods of the Class:

Method Name	Parameters	Return Type	Description
getGroupName	void	String	Retrieves groupName attribute
setGroupName	groupName: *string*	Void	Assigns groupName attribute
getGroupID	void	Integer	Retrieves groupID attribute
setGroupID	groupID: *integer*	Void	Assigns groupID attribute
getPermitted DownloadSize	void	Integer	Retrieves permittedDownloadSize attribute
setPermitted DownloadSize	permitted DownloadSize: *integer*	Void	Assigns permittedDownloadSize attribute

UserInGroup Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
IP	char(15)	The IP of the user's computer
groupID	integer	A unique integer which specifies the user group

Methods of the Class:

Method Name	Parameters	Return Type	Description
addUserToGroup	IP: *char(15)* groupID: *integer*	Void	Adds new user who has the specified IP into the group with specified groupID
deleteUserFrom Group	IP: *char(15)* groupID: *integer*	void	Deletes the existing users from the user groups
getUsersInGroup	groupID: *integer*	IP list	Retrieves the IP's of the users in the specified user group

PacketParser Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
header	string	The header part of an IP packet
data	string	The data part of an IP packet

Methods of the Class:

Method Name	Parameters	Return Type	Description
getPacket	void	string	Reads the packet information from the socket and returns that information in a string
parseHeader	NetworkTraffic Log object	void	Parses the header part of an IP packet in order to get the information about sourceIP, destinationIP, and packetSize
parseData	void	string	Parses the data part of an IP packet in order to get the accessed URL information

NetworkTrafficLog Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
sourceIP	char(15)	The source sending the packet
destinationIP	char(15)	The destination of the packet
packetSize	integer	The size of the packet
time	string	Time of the communication
accessedURL	string	The destination URL

Methods of the Class:

Method Name	Parameters	Return Type	Description
getSourceIP	void	char(15)	Retrieves SourceIP attribute
setSourceIP	sourceIP: *char(15)*	Void	Assigns SourceIP attribute
getDestinationIP	void	char(15)	Retrieves DestinationIP attribute
setDestinationIP	destinationIP: *char(15)*	void	Assigns DestinationIP attribute
getPacketSize	void	integer	Retrieves PacketSize attribute
setPacketSize	packetSize: *integer*	void	Assigns PacketSize attribute
getTime	void	string	Retrieves Time attribute
setTime	time: *string*	void	Assigns Time attribute
getAccessedURL	void	string	Retrieves AccessedURL attribute
setAccessedURL	accessedURL: *string*	void	Assigns AccessedURL attribute

MonitoringScreen Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
displayCurrentTraffic	void	Network TrafficLog	Displays the incoming and outgoing web traffic on the web site

ObserveNetworkStatisticsScreen Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
URL	string	The URL for displaying hit rate statistics
hitRate	string	The hit rate of the URL specified in URL attribute
packetSize	integer	The size of the packet
time	string	Time of the communication
accessedURL	string	The destination URL, for URL request statistics
sourceIP	char(15)	The source IP which sends the packet
destinationIP	char(15)	The destination IP which will fetch the packet
userGroupName	string	The name of the user group for URL request statistics

Methods of the Class:

Method Name	Parameters	Return Type	Description
computeHitRates	specifiedTime Interval: *string* groupName: *string*	void	For the given group name (optional) and the specified time interval, this method calculates and sets the hitRate and URL attributes.
computeURLRequests ForUser	specifiedTime Interval: *string* IP: *char(15)*	void	For the given IP and the specified time interval, this method retrieves and sets the sourceIP, destinationIP, accessedURL, time and packetSize attributes.
computeDownload InformationForUser	specifiedTime Interval: *string* IP: *char(15)*	void	For the given IP and the specified time interval, this method retrieves and sets the time attribute.
computeNetwork Traffic	specifiedTime Interval: *string* groupName: *string*	void	For the given user group (optional) and the specified time interval, this method retrieves and sets the userGroupName, destinationIP, accessedURL, time and packetSize attributes.
computeConfidential DataViolation	specifiedTime Interval: *string* IP: *char(15)*	Confidential Data	For the given IP (optional) and the specified time interval (optional), this method retrieves and sets the sourceIP, destinationIP, time and accessedURL attributes.
computeFilteredContent	specifiedTime Interval: *string* IP: *char(15)*	Filtered Content	For the given IP (optional) and the specified time interval (optional), this method retrieves and sets the sourceIP, destinationIP, time and accessedURL attributes
displayComputed Statistics	void	void	Displays the computed statistics on the web site.

SystemManagementScreen Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
displayRunningMode	void	void	Displays the current running mode of the system.
displayBlackURLGroup	group: *BlackURLgroup*	void	Displays the URLs in the given black URL group.
displayUpdateMessage	void	void	Displays a success or failure message on the screen.
displayBlackWordGroup	group: *BlackWordgroup*	void	Displays the words in the given black word group.
displayBlackURLList	void	void	Displays the black URLs.
displayWhiteURLList	void	void	Displays the white URLs.
displayBlackWordList	void	void	Displays the black words.
displayWhiteWordList	void	void	Displays the white words.
displayBayesWords	void	void	Displays the words and their occurrences used by the Bayesian algorithm.
updateRunningMode	string	boolean	Updates the current running mode of the system.
updateBlackURLGroup	group: *BlackURLgroup*	boolean	Updates the URLs in the given black URL group.
updateBlackWordGroup	group: *BlackWordgroup*	boolean	Updates the words in the given black word group.
updateBlackURLList	URL: *BlackURL*	boolean	Updates the black URLs.
updateWhiteURLList	URL: *WhiteURL*	boolean	Updates the white URLs.
updateBlackWordList	word: *BlackWord*	boolean	Updates the black words.
updateWhiteWordList	word: *WhiteWord*	boolean	Updates the white words.
specifyConfidentialData Keyword	keyword: *string*	void	Specifies the confidential to be added, deleted, or updated.
specifyConfidentialData Priority	priority: *string*	void	Specifies the criticality of the confidential data.

RunningMode Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
Mode	string	The current running mode of the system (free mode, normal mode or secure mode)

Methods of the Class:

Method Name	Parameters	Return Type	Description
getMode	void	string	Retrieves mode attribute
setMode	mode: *string*	void	Assigns mode attribute

BlackWord Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
wordID	int	Each black word is assigned an ID to ease the use of it
word	string	The black word that is to be dealt with
isActive	boolean	Word can be active or inactive according to the isActive attribute

Methods of the Class:

Method Name	Parameters	Return Type	Description
setWord	word: *string*	void	Assigns specified black word to the word attribute of the class
setIsActive	boolean	void	Assigns the boolean value of the isActive attribute of the class
showGroupsOfWord	void		Retrieves the groups which the word is in, and displays the groups
getWord	void	string	Retrieves the word

BlackWordGroup Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
groupID	integer	The ID of the group
groupName	string	The descriptive name of the group
isActive	boolean	Whether this setting is active currently

Methods of the Class:

Method Name	Parameters	Return Type	Description
get groupID	void	integer	Retrieves groupID attribute
set groupID	groupID: *integer*	void	Assigns groupID attribute
get groupName	void	string	Retrieves groupName attribute
set groupName	groupName: *string*	void	Assigns groupName attribute
getIsActive	void	boolean	Retrieves isActive attribute
setIsActive	isActive: *boolean*	void	Assigns isActive attribute

WordInGroup

Attributes of the Class:

Attribute Name	Attribute Type	Description
wordID	integer	A unique integer which specifies the word
groupID	integer	A unique integer which specifies the word group

Methods of the Class:

Method Name	Parameters	Return Type	Description
addWordToGroup	wordID: *integer* groupID: *integer*	void	Adds new word, which has the specified wordID, into the group with specified groupID
deleteWordFrom Group	wordID: *integer* groupID: *integer*	void	Deletes the existing words from the word groups

BlackURL Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
URLID	integer	Unique integer which specifies the black URLs
URL	string	blackURL information
isActive	boolean	Activation flag of the URL

Methods of the Class:

Method Name	Parameters	Return Type	Description
setURL	string	void	Sets the URL attribute to the parameter string
setIsActive	boolean	void	Sets the isActive attribute of the class
getURL	void	string	Retrieves the URL attribute of the class and returns it in a string
showGroupsOf URL	void	BlackURL GroupList	Retrieves the groups that the URL belongs to

BlackURLGroup Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
groupID	int	Each group is assigned an ID to ease the use of it
groupName	string	The name of the group
isActive	boolean	Defines if the group is active or inactive

Methods of the Class:

Method Name	Parameters	Return Type	Description
getGroupName	void	string	Retrieves the name of the group
setGroupName	groupName: *string*	void	Assigns the value of the groupName attribute
setIsActive	boolean	void	Assigns the boolean value of the isActive attribute
showURLsInGroup	void		Retrieves the URLs which belongs to the group and display them

WhiteWord Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
worded	integer	Unique integer which specifies the white words
Word	string	White word information
isActive	boolean	Activation flag of the white word

Methods of the Class:

Method Name	Parameters	Return Type	Description
setWord	string	void	Sets the word attribute of the class to the parameter string
setIsActive	boolean	void	Sets the isActive attribute of the class
getWord	void	string	Retrieves the word attribute of the class and returns it in a string

WhiteURL Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
URLID	integer	The ID of the URL
URL	string	The URL
isActive	boolean	Whether this setting is active currently

Methods of the Class:

Method Name	Parameters	Return Type	Description
get URLID	void	integer	Retrieves URLID attribute
set URLID	URLID: *integer*	void	Assigns URLID attribute
get URL	void	string	Retrieves URL attribute
set URL	URL: *string*	void	Assigns URL attribute
getIsActive	void	boolean	Retrieves isActive attribute
setIsActive	isActive: *boolean*	void	Assigns isActive attribute

ConfidentialData Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
data	string	Confidential data to be prevented to go outside the local area network
criticality	integer	The criticality of the confidential data

Methods of the Class:

Method Name	Parameters	Return Type	Description
setData	data: *string*	void	Assigns the value of the data attribute
getData	void	string	Retrieves the value of the data attribute
setCriticality	criticality: *integer*	void	Assigns the value of the criticality attribute
getCriticality	void	integer	Retrieves the value of the criticality attribute

FilteredContent Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
data	string vector	Forbidden content that caused the filtering of the packet
time	string	Time of the communication
sourceIP	char(15)	Source IP of the communication
destinationIP	char(15)	Destination IP of the communication
URL	String	Accessed URL address

Methods of the Class:

Method Name	Parameters	Return Type	Description
setData	data: *string*	void	Assigns the values of the data attribute
getData	void	string	Retrieves the values of the data attribute
setTime	time: *string*	void	Assigns the values of the time attribute
getTime	void	string	Retrieves the values of the time attribute
setSourceIP	sourceIP: *char(15)*	void	Assigns the values of the sourceIP attribute
getSourceIP	void	char(15)	Retrieves the values of the sourceIP attribute
setDestinationIP	destinationIP: *char(15)*	void	Assigns the values of the destinationIP attribute
getDestinationIP	void	char(15)	Retrieves the values of the destinationIP attribute
setURL	URL: *string*	void	Assigns the values of the URL attribute
getURL	void	string	Retrieves the values of the URL attribute

ConfidentialDataViolations Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
userIP	char(15)	The IP sending confidential data violating packet
violatedRuleID	integer	The identification number of the confidential data
time	string	Time of the violation

Methods of the Class:

Method Name	Parameters	Return Type	Description
setUserIP	userIP: *char(15)*	void	Assigns the values of the userIP attribute
getUserIP	void	char(15)	Retrieves the values of the userIP attribute
setTime	time: *string*	void	Assigns the values of the time attribute
getTime	void	string	Retrieves the values of the time attribute
setViolatedRuleID	violatedRuleID: *integer*	void	Assigns the values of the violatedRuleID attribute
getViolatedRuleID	void	integer	Retrieves the values of the violatedRuleID attribute

BayesWord Class:

Attributes of the Class:

Attribute Name	Attribute Type	Description
wordID	integer	The identification number of the word
word	string	The word to be inspected
maliciousOccurrences	integer	The number of occurrences of the word in malicious site packets
harmlessOccurrences	integer	The number of occurrences of the word in harmless site packets

Methods of the Class:

Method Name	Parameters	Return Type	Description
setWordID	wordID: *char(15)*	void	Assigns the values of the wordID attribute
getWordID	void	char(15)	Retrieves the values of the wordID attribute
setWord	word: *string*	void	Assigns the values of the word attribute
getWord	void	string	Retrieves the values of the word attribute
setMalicious Occurrences	malicious Occurences: *integer*	void	Assigns the values of the maliciousOccurrences attribute
getMalicious Occurrences	void	integer	Retrieves the values of the maliciousOccurences attribute
setHarmless Occurrences	harmless Occurrences: *integer*	void	Assigns the values of the harmlessOccurrences attribute
getHarmless Occurrences	void	integer	Retrieves the values of the harmlessOccurrences attribute

URLParser Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
fetchURLContent	URL: *string*	void	Fetches the content of the URL, that is specified by the administrator
evaluateBayesWords	word: *string*	integer	Evaluates the maliciousOccurences and harmlessOccurrences of a word in the specified URL content

LearningActivationScreen Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
getURL	URL: *string*	void	Gets the specified URL from the administrator
getURLType	word: *string*	integer	Gets the specified URL's type from the administrator (malicious vs. harmless URL)

Control Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
controlBlackWord	word: *string*	boolean	Checks whether the black words exist in the incoming packet
controlWhiteWord	word: *string*	boolean	Checks whether the white words exist in the incoming packet
reportFiltering	FilteredContent object	void	Saves the associated information of the filtered packets

Control2 Class:

Methods of the Class:

Method Name	Parameters	Return Type	Description
controlConfidentialData	word: *string*	boolean	Applies the confidential data detection algorithm to the outgoing packets
reportViolation	Confidential DataViolations object	void	Saves the associated information of the packets that include confidential data

Control3 Class:

Methods of the Class:

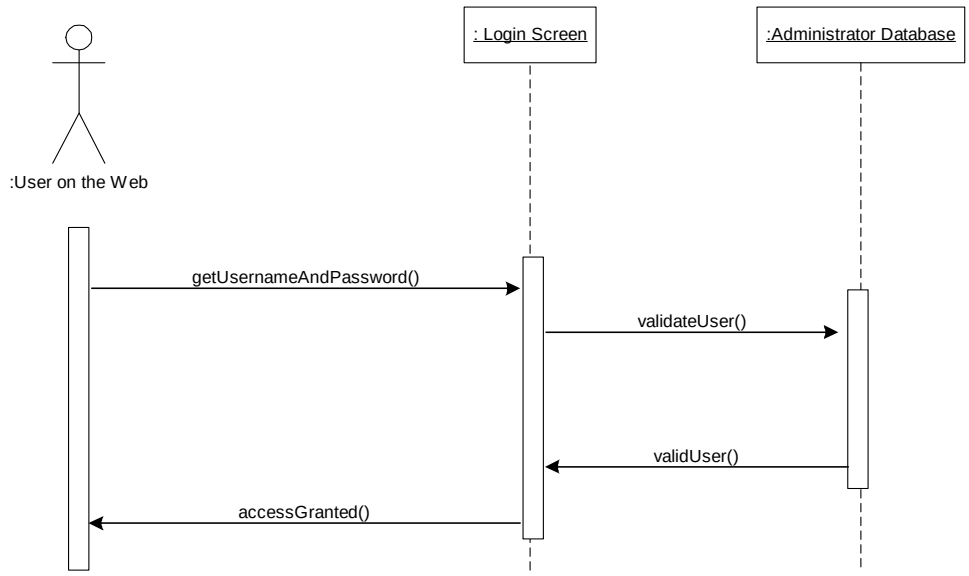
Method Name	Parameters	Return Type	Description
applyBayesianAlgoritm	void	boolean	Applies the Bayesian algorithm to the words of the incoming packets
reportFiltering	FilteredContent object	void	Saves the associated information of the filtered packets

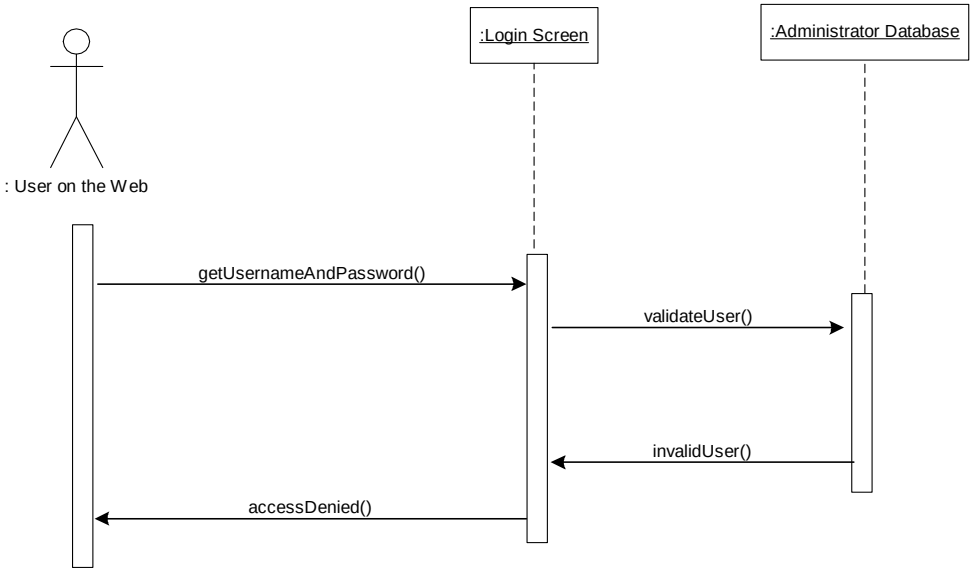
3.3 Sequence Diagrams

3.3.1 System Management Module

Under the System Management Module, we have only presented the authentication sequence diagram, but it is also responsible for the user interface of the other modules.

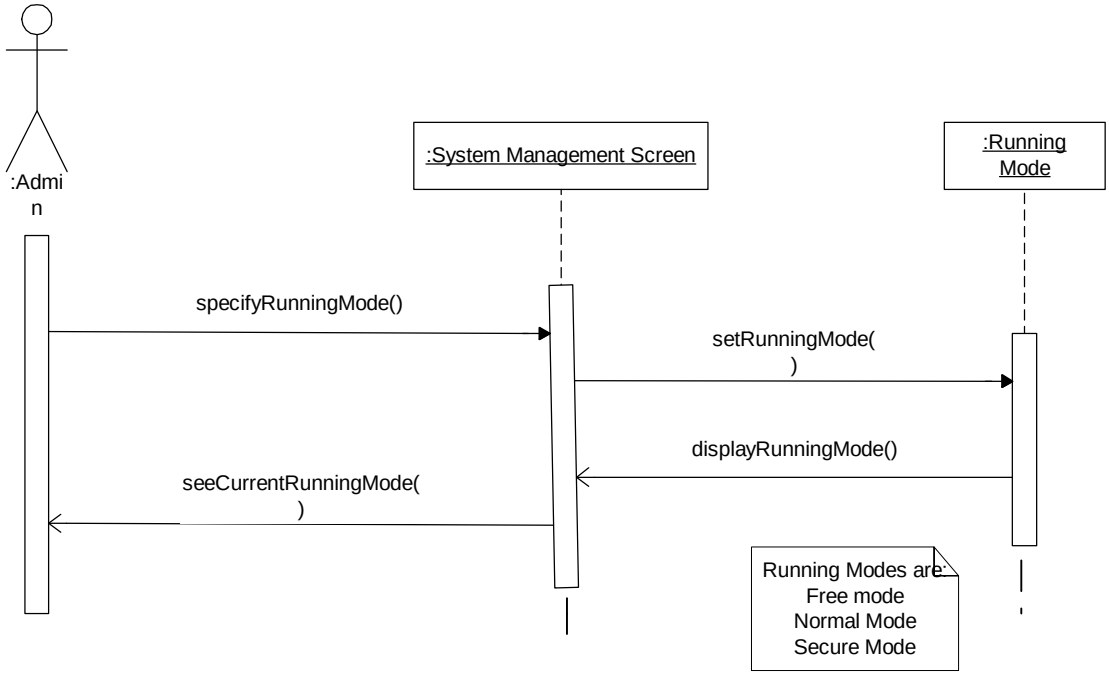
3.3.1.1 Authentication on the Web





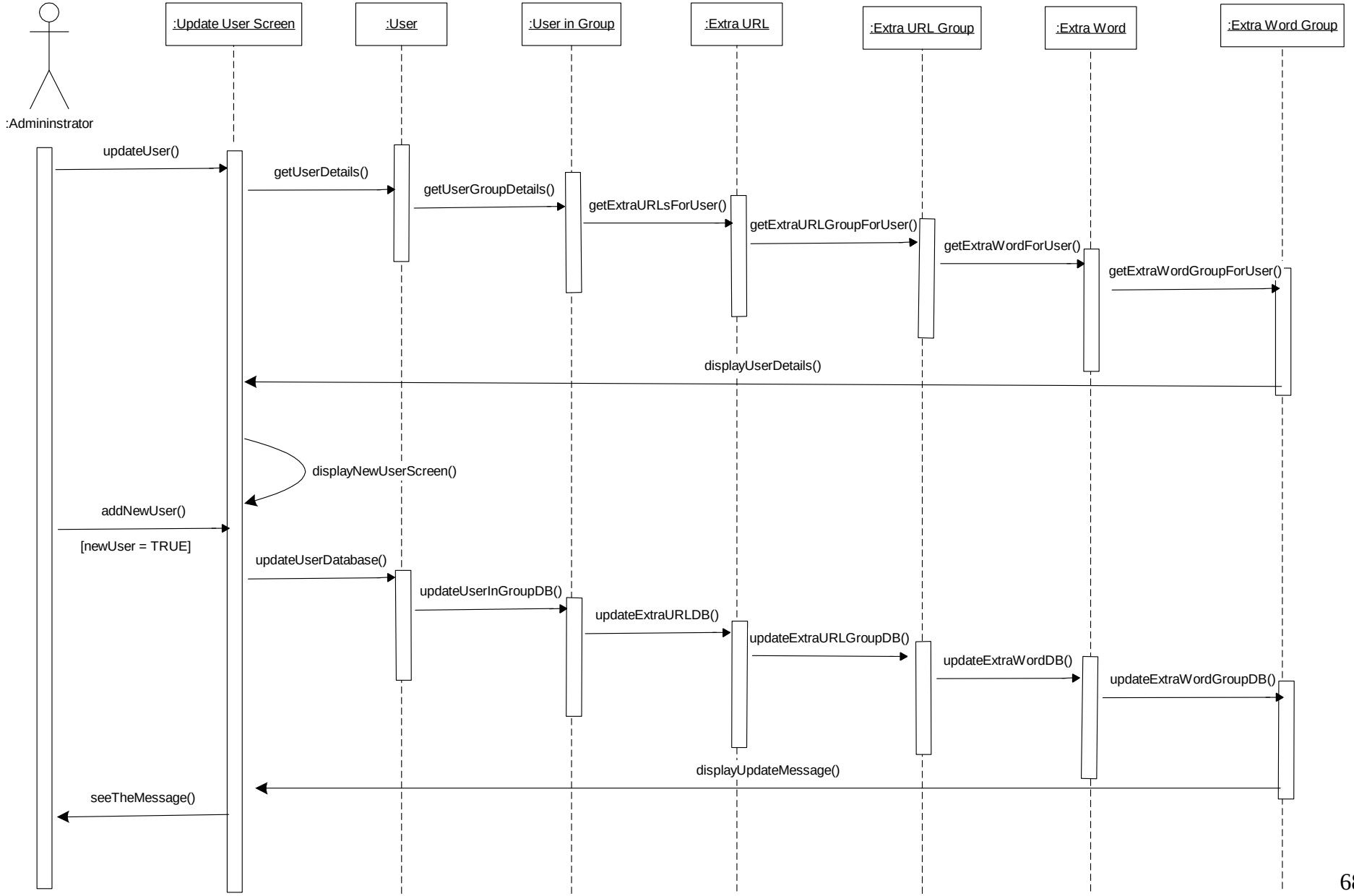
Sequence of Events for Authentication on the Web:	
Main Sequence	1- The administrator must log into the system in order to perform the system management facilities. To log into the system, he/she enters his/her username and password to the related fields in the Login Screen. 2- The username and password of the administrator will be checked by the Administrator database table. If these fields match with the fields in the Administrator table, then access granted. Otherwise access will be denied.

3.3.1.2 Specify the Running Mode of the System



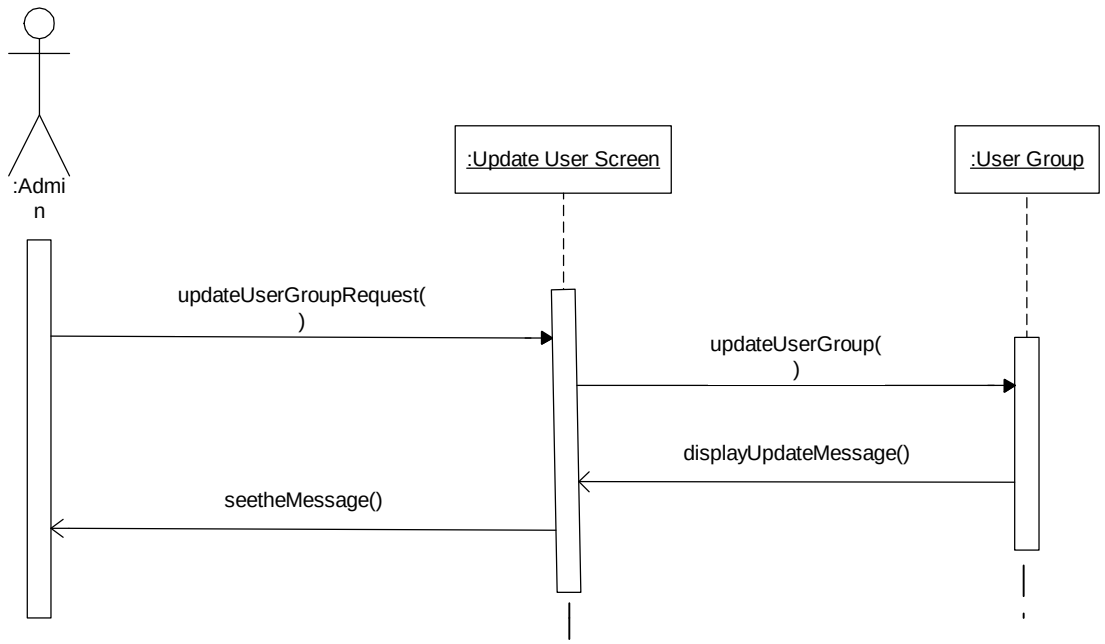
Sequence of Events for Specifying the Running Mode of the System:	
Main Sequence	1- After administrator logs into the system, the system management screen will be displayed. He/she can set the running mode of the system by choosing among free mode, normal mode and secure mode. 2- Then the specified running mode will be written to the runningMode table in the database.

3.3.1.3 Update Users of the System



Sequence of Events for Updating Users of the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/delete/update user details. (For readability, all these tasks are represented as “update” in the Sequence Diagrams.)
Alternative Sequence	3- Administrator chooses to add a user. 4- The screen for adding user details is displayed. 5- Administrator specifies the user details (Identification details, group details, extra URL and word restrictions for the user). 6- Insertions are written to the database tables. (User, UserInGroup, ExtraURL, ExtraURLGroup, ExtraWord, ExtraWordGroup) 7- A success or failure message is displayed.
Alternative Sequence	3- Administrator chooses to update a user. 4- The screen for updating user details is displayed. 5- Administrator specifies the user by providing his / her IP. 6- If the user exists in the database, his/her details will be retrieved and displayed on the screen. 7- Administrator makes the necessary updates. 8- The updates are written to the database tables. 9- A success or failure message is displayed.
Alternative Sequence	3- Administrator chooses to delete a user. 4- The screen for deleting a user is displayed. 5- Administrator specifies the user by providing his / her IP. 6- If the user exists in the database, the user and his/her associated details are deleted from the database tables. 7- A success or failure message is displayed.

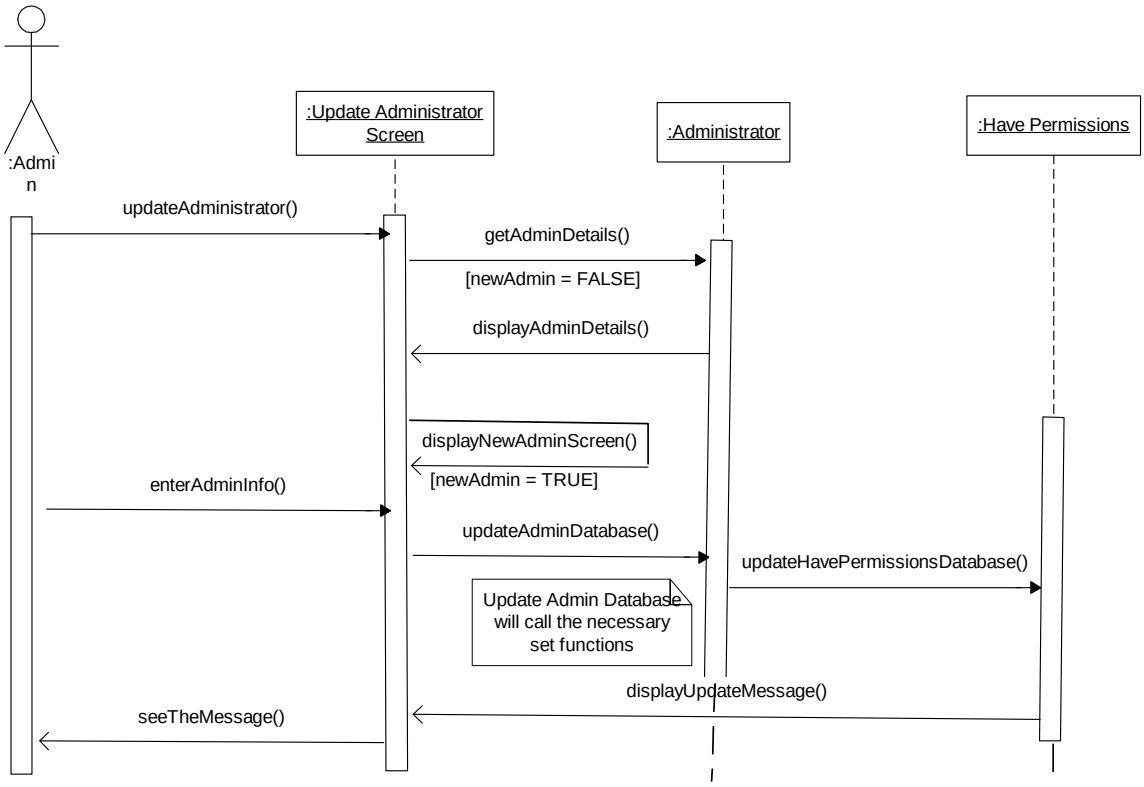
3.3.1.4 Update User Groups of the System



Sequence of Events for Updating User Groups of the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/delete/update user group details. (For readability, all these tasks are represented as “update” in the Sequence Diagrams.)
Alternative Sequence	3- Administrator chooses to add a user group. 4- The screen for adding a user group is displayed. 5- Administrator specifies the user group details. 6- Insertions are written to the database tables. (UserGroup). 7- A success or failure message is displayed.

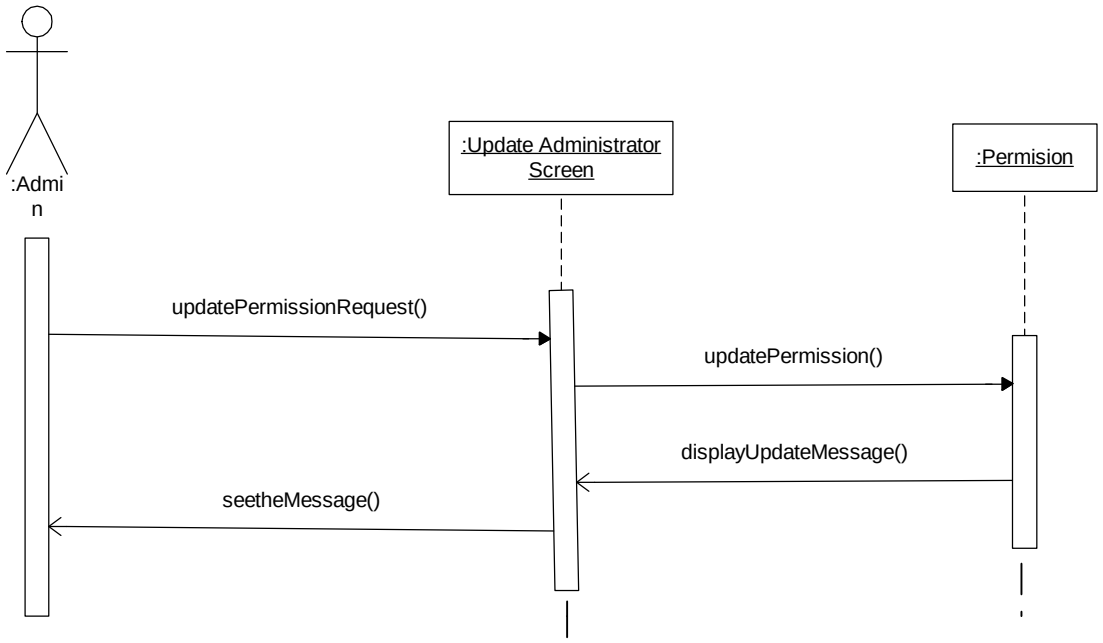
Alternative Sequence	3- Administrator chooses to update a user group. 4- The screen for updating user group is displayed. 5- Administrator specifies the user group by providing its name. 6- If the user group exists in the database, its details will be retrieved and displayed on the screen. 7- Administrator makes the necessary updates. 8- The updates are written to the database tables. 9- A success or failure message is displayed.
Alternative Sequence	3- Administrator chooses to delete a user group. 4- The screen for deleting a user group is displayed. 5- Administrator specifies the user group by providing its name. 6- If the user group exists in the database, the user group will be deleted from the database tables. 7- A success or failure message is displayed.

3.3.1.5 Update Administrators of the System



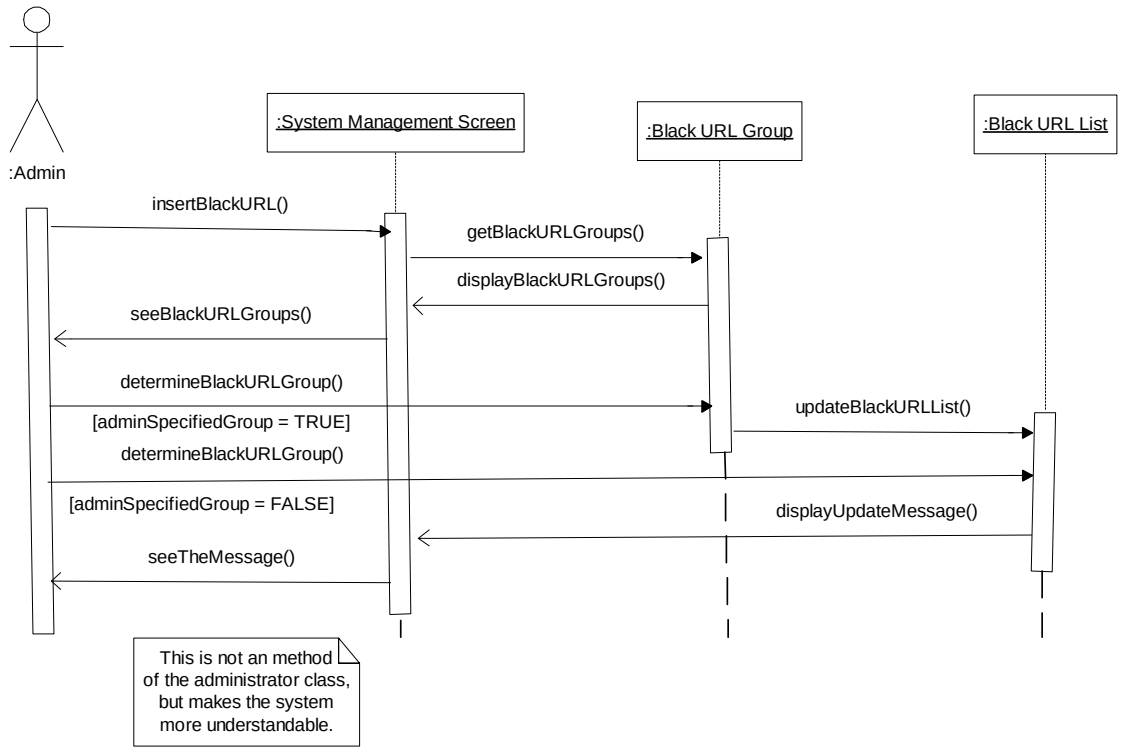
Sequence of Events for Updating Administrators of the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/delete/update administrator details.
Alternative Sequence	3- Administrator chooses to add an administrator. 4- The screen for adding administrator is displayed. 5- Administrator specifies the administrator details (Identification details and permissions). 6- Insertions are written to the database tables. (Administrator and HavePermissions) 7- A success or failure message is displayed.

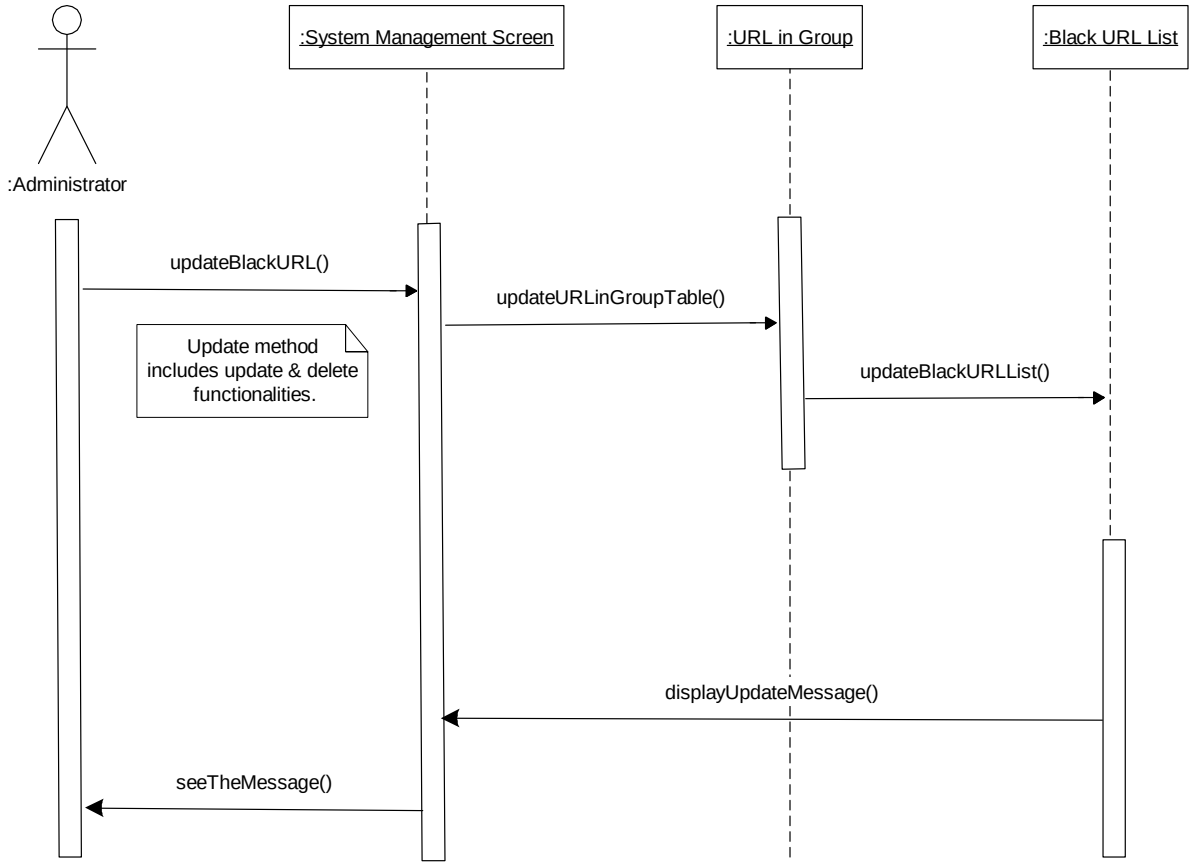
Alternative Sequence	3- Administrator chooses to update an administrator. 4- The screen for updating administrator is displayed. 5- Administrator specifies the administrator by providing his / her user name. 6- If the administrator exists in the database, his/her details will be retrieved and displayed on the screen. 7- Administrator makes the necessary updates. 8- The updates are written to the database tables. 9- A success or failure message is displayed.
Alternative Sequence	3- Administrator chooses to delete an administrator. 4- The screen for deleting an administrator is displayed. 5- Administrator specifies the administrator by providing his / her user name. 6- If the administrator exists in the database, the administrator and his/her associated details are deleted from the database tables. 7- A success or failure message is displayed.



Sequence of Events for Updating Administrator Permissions Defined in the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/delete/update administrator permissions defined in the system. 3- Administrator chooses to add/delete/update permission. 4- The screen for updating permissions is displayed. 5- Administrator makes the change in the permission. 6- Updates are written to the database tables. (Permission) 7- A success or failure message is displayed.

3.3.1.6 Update Black URL List of the System

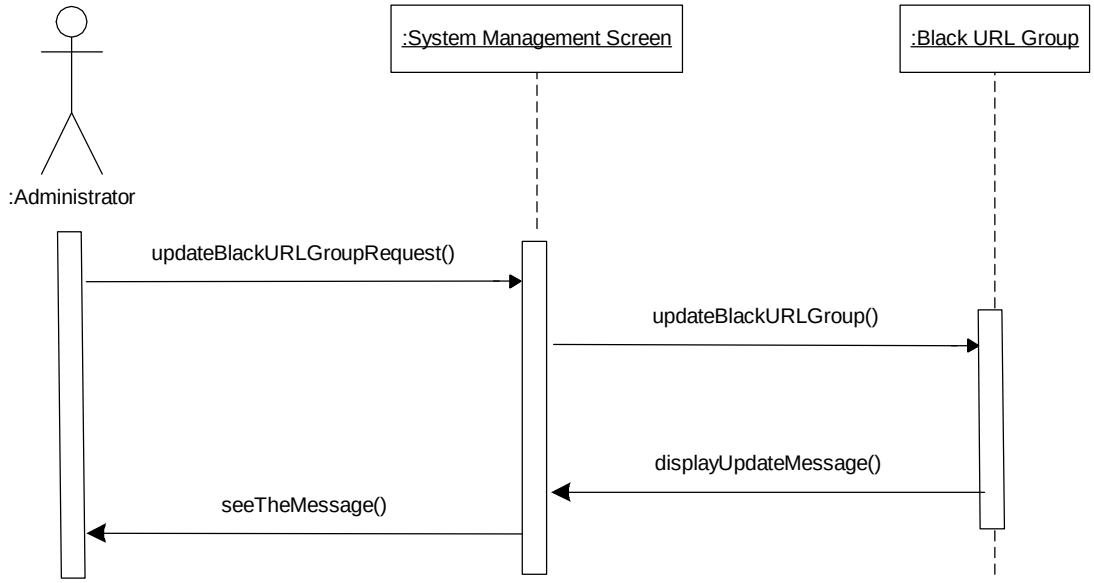




Sequence of Events for Updating Black URL List of the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/update/delete a black URL.
Alternative Sequence	3- Administrator chooses to insert a black URL. 4- The screen for inserting a black URL is displayed. Also, the black URL groups currently available are retrieved from the database and displayed on the screen. 5- Administrator provides the black URL. 6- Insertion is written to the database table. (BlackURLList) 7- Administrator has the option of specifying the group of the black URL. 8- Insertion is written to the database table. (URLInGroup) 9- A success or failure message is displayed.
Alternative Sequence	3- Administrator chooses to update a black URL. 4- The screen for updating a black URL is displayed. Also, the black URL groups currently available are retrieved from the database and displayed on the screen. 5- Administrator makes the change. 6- Update is written to the database table. (BlackURLList) 7- Administrator has the option of updating the groups of the black URL. 8- Update is written to the database table. (URLInGroup) 9- A success or failure message is displayed.

Alternative Sequence	3- Administrator chooses to delete a black URL. 4- The screen for deleting a black URL is displayed. 5- Administrator specifies the URL to be deleted. 6- Black URL and associated details is deleted from the database table. (BlackURLList and URLInGroup) 7- A success or failure message is displayed.

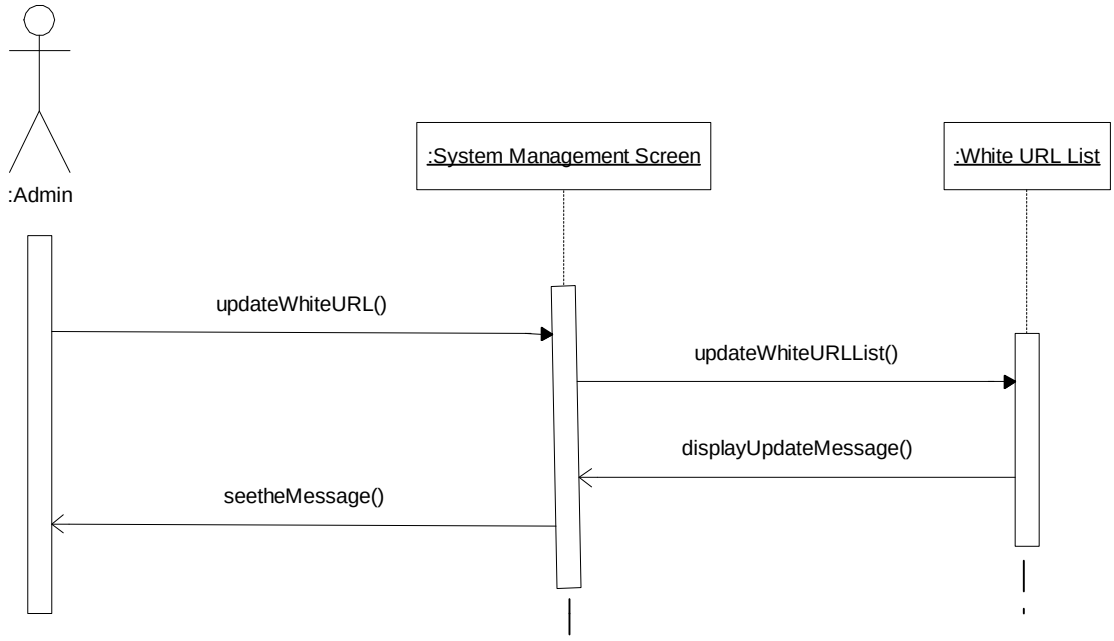
3.3.1.7 Update Black URL Groups of the System



Since the above tables show the flow of control in add, update and delete requests explicitly, we will not duplicate the alternative sequence in the following tables. The main sequence will represent the possible alternative flows.

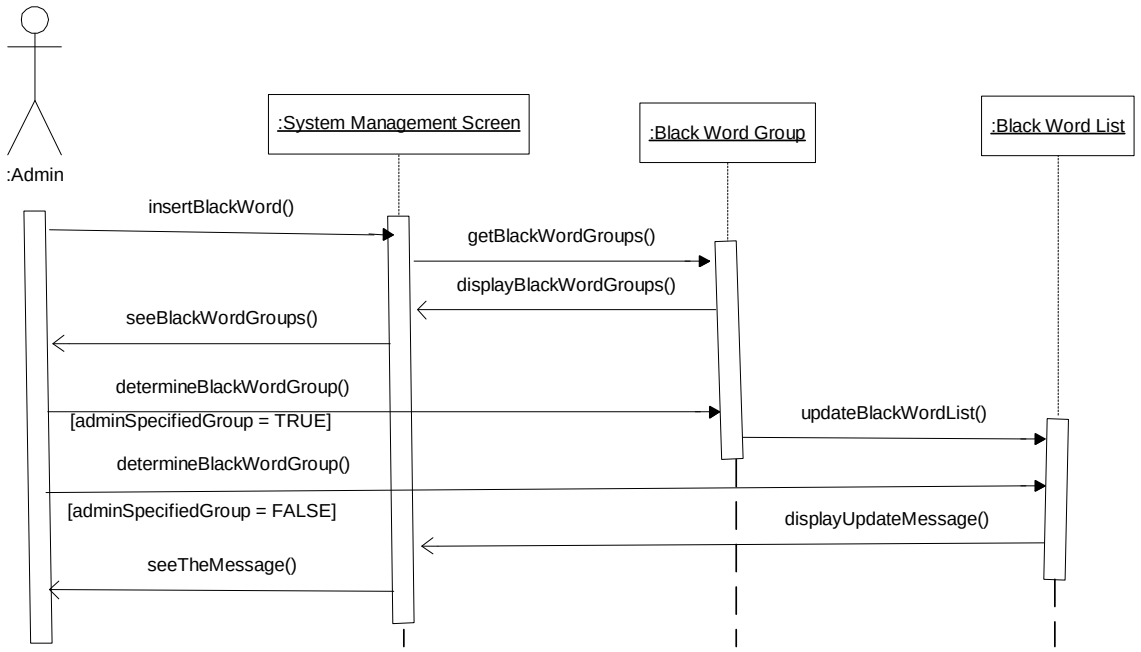
Sequence of Events for Updating Black URL Groups of the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/delete/update black URL group details. (For readability, all these tasks are represented as “update” in the Sequence Diagrams.) 3- Administrator chooses to add/delete/update a black URL group. 4- The screen for add/delete/update a black URL group is displayed. 5- Administrator specifies the change to be done in the database. 6- Changes are written to the database tables. (BlackURLGroup). 7- A success or failure message is displayed.

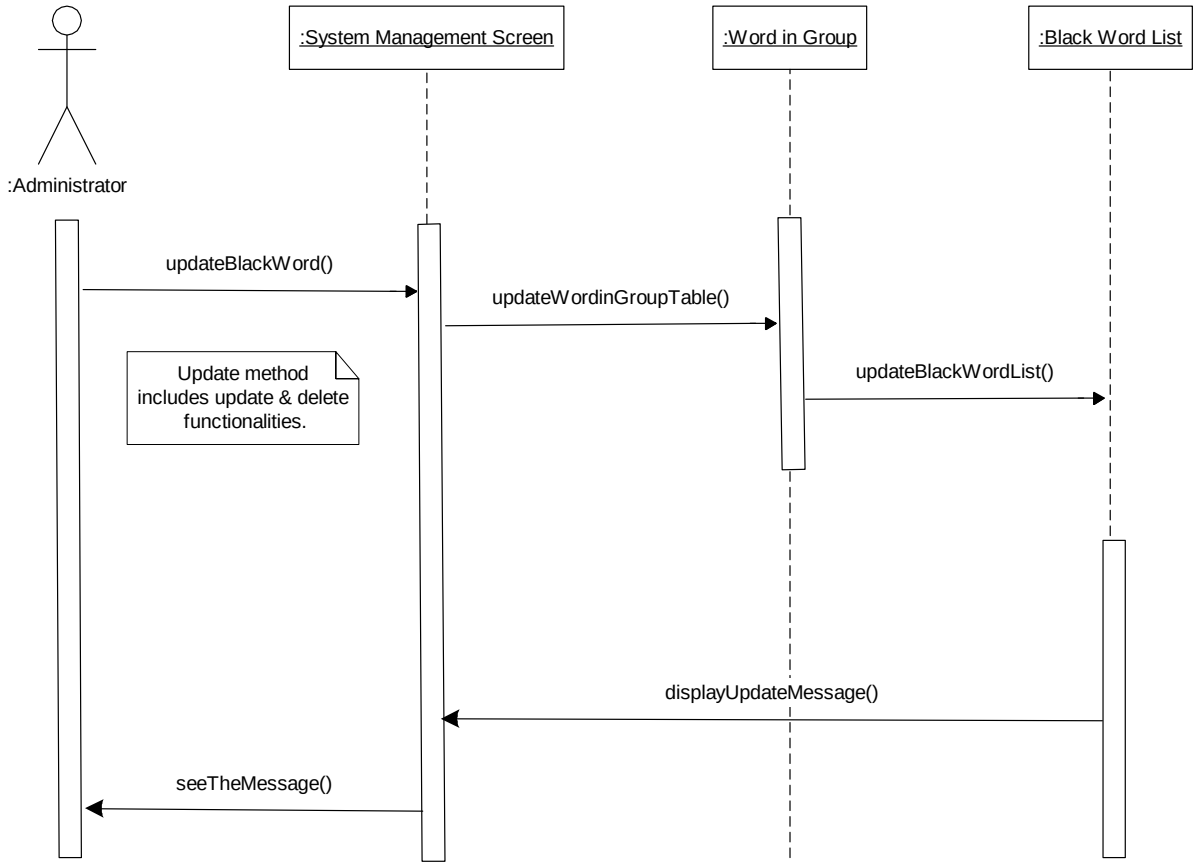
3.3.1.8 Update White URL List of the System



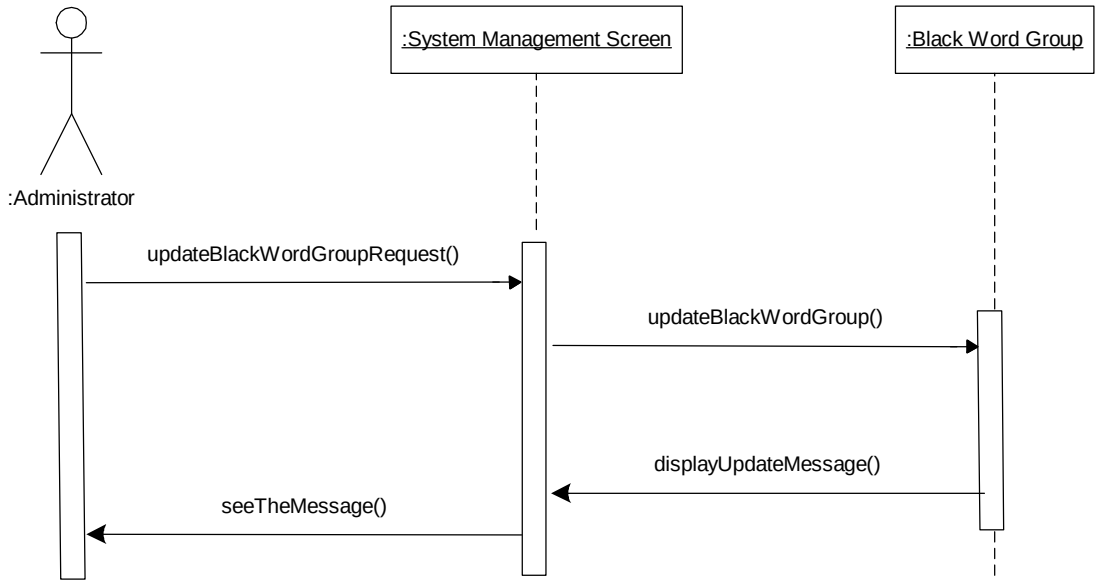
Sequence of Events for Updating White URL List of the System:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to add/delete/update a white URL. (For readability, all these tasks are represented as “update” in the Sequence Diagrams.) 3- Administrator chooses to add/delete/update a white URL. 4- The screen for add/delete/update a white URL is displayed. 5- Administrator specifies the change to be done in the database. 6- Changes are written to the database tables. (WhiteURLList). 7- A success or failure message is displayed.

3.3.1.9 Update Black Word List of the System

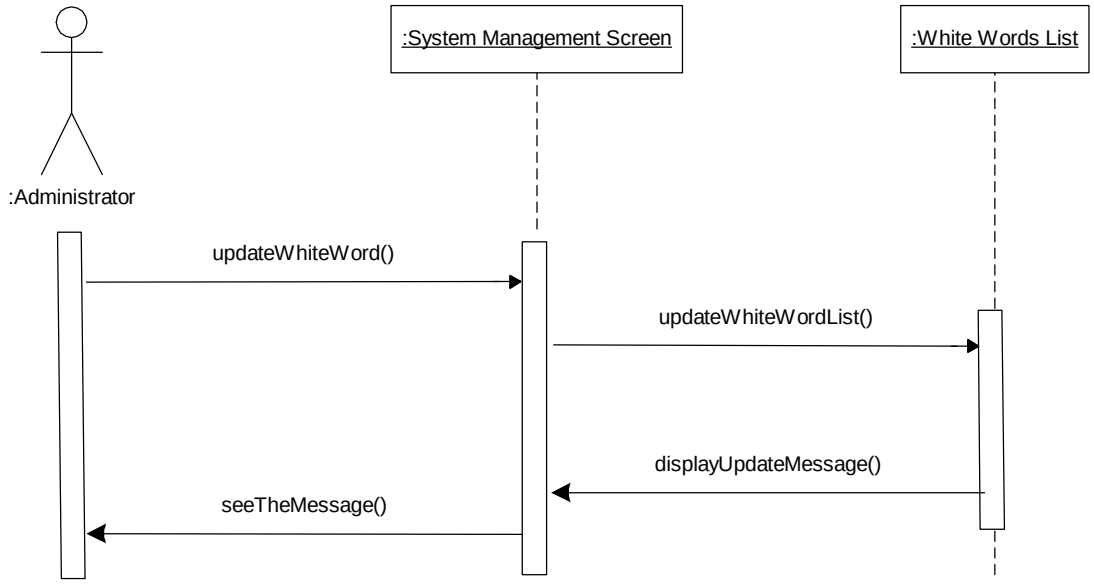




3.3.1.10 Update Black Word Groups of the System

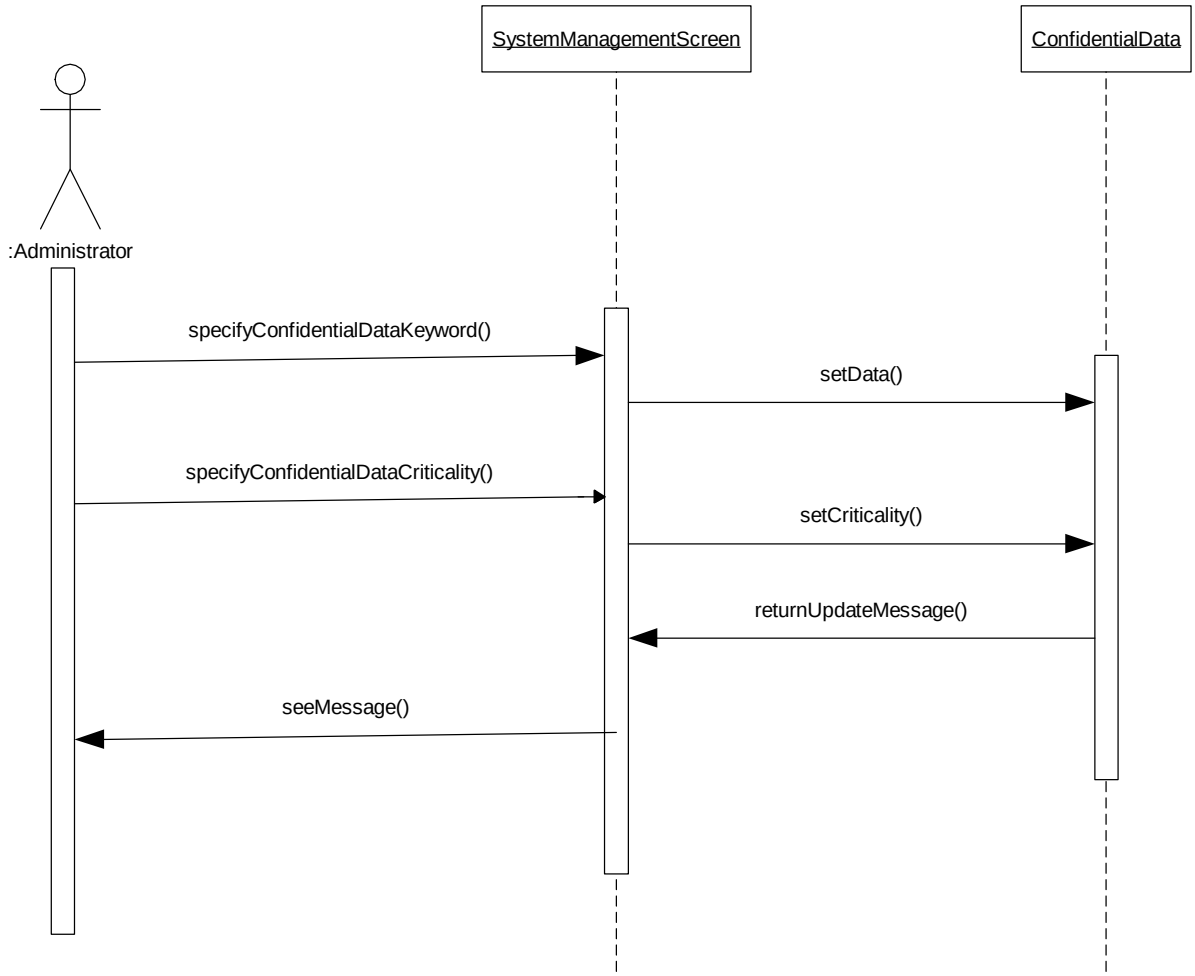


3.3.1.11 Update White Word List of the System



The sequences concerning black and white words follow exactly the same flow with the black and white URLs. Therefore, sequence descriptions of the *Update Black Word List of the System*, *Update Black Word Groups* and *Update White Word List of the System* diagrams will be skipped here.

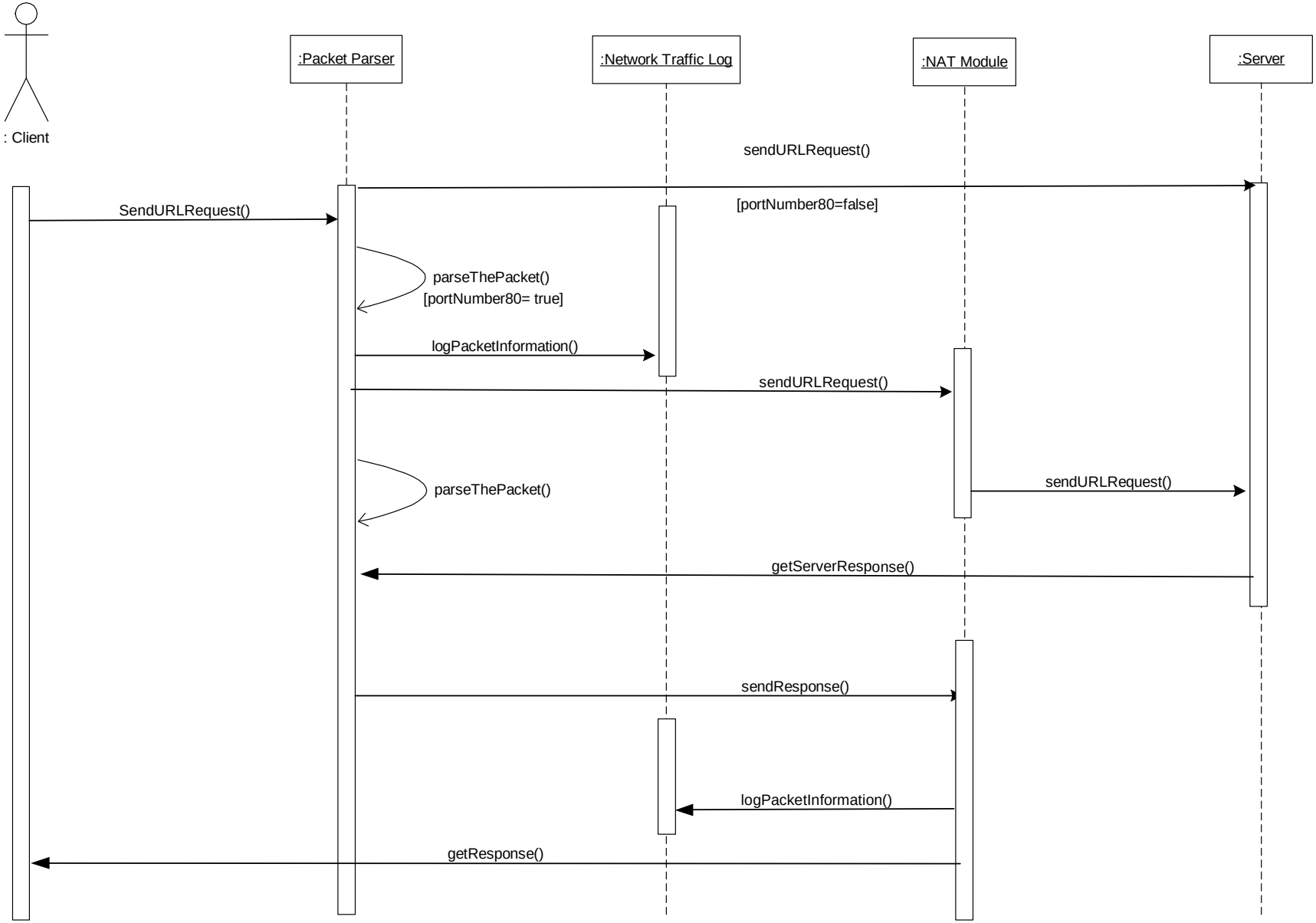
3.3.1.12 Specify Confidential Data to Be Protected



Sequence of Events for Specifying Confidential Data to Be Protected:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator has the option of entering new confidential data and selects its priority from a drop-down list, setting previously specified confidential data as not active or changing previously specified confidential data's priority. 3- The ConfidentialData table will be set accordingly.

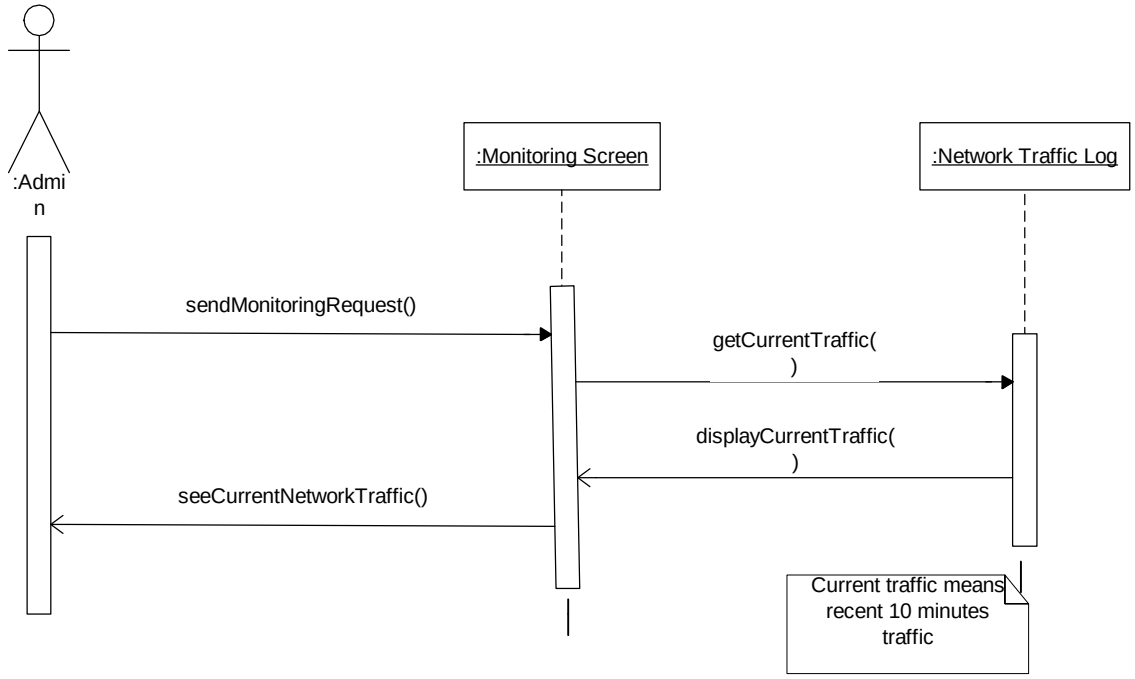
3.3.2 Network Traffic Monitoring Module

3.3.2.1 Saving the Network Traffic Logs



Sequence of Events for Saving the Network Traffic Logs:	
Main Sequence	1- The packets will be caught by the system and inspected to see if they are associated with port number 80. 2- In case they are, they will be parsed and necessary information will be acquired. 3- Packet information, source and destination IP's, accessed URL, time and size information will be logged in the NetworkTrafficLog table in the database.

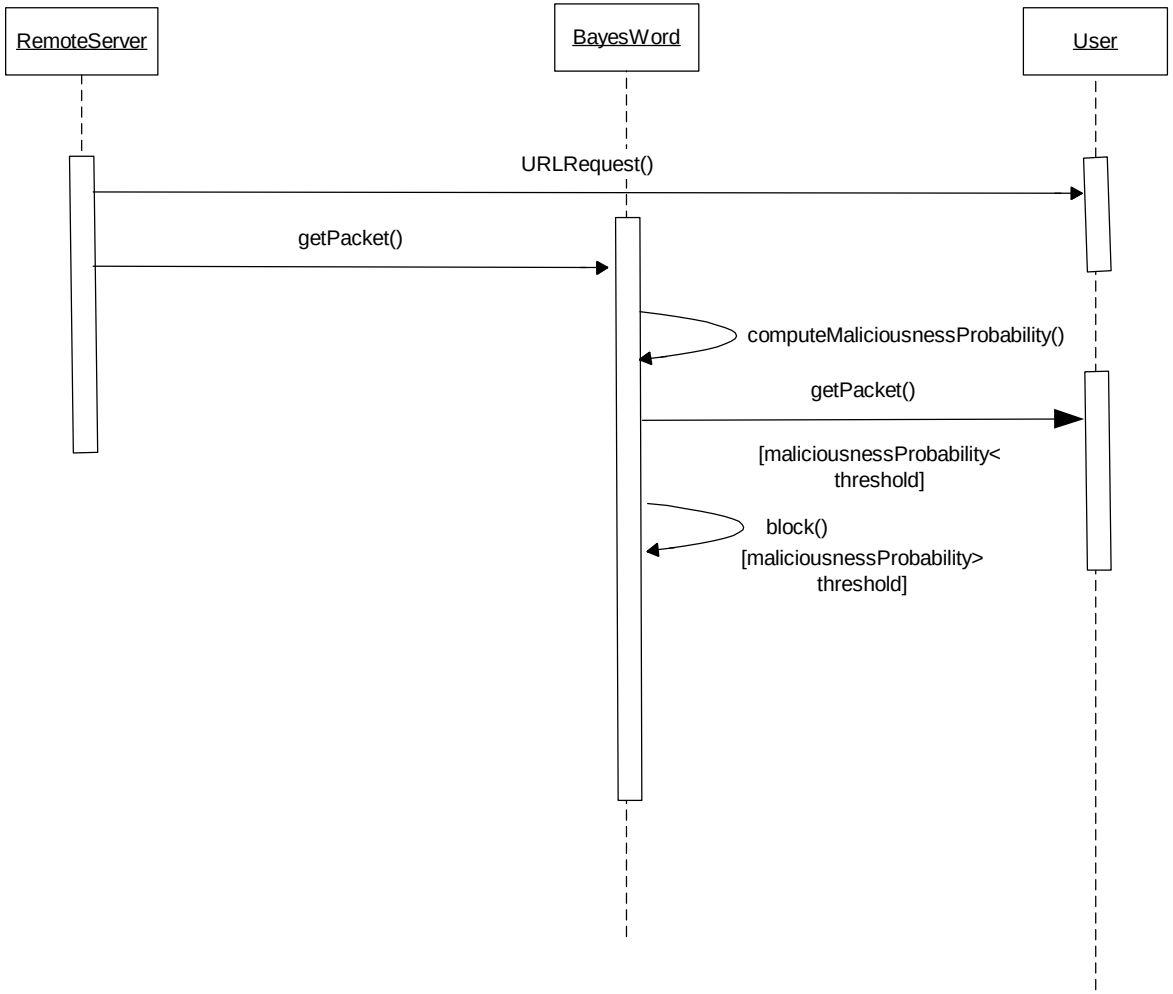
3.3.2.2 Monitoring Network Traffic



Sequence of Events for Viewing Network Traffic in Real Time:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to view the network traffic in real time. 3- The source IPs, destination IPs, accessed URLs, communication size and time information of incoming and outgoing packets will be displayed.

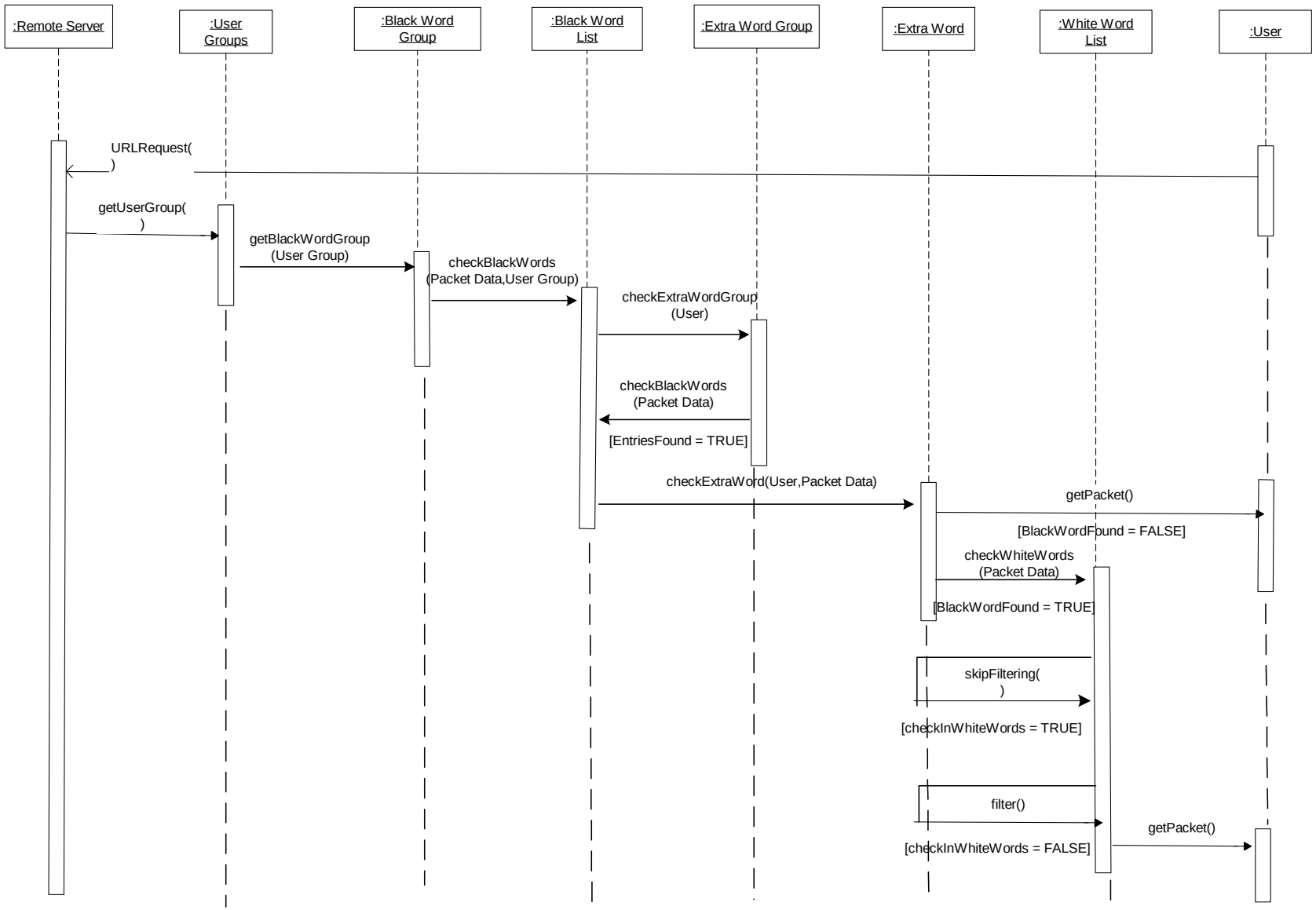
3.3.3 Content Filtering Module

3.3.3.1 Applying Content Filtering (1st Algorithm)



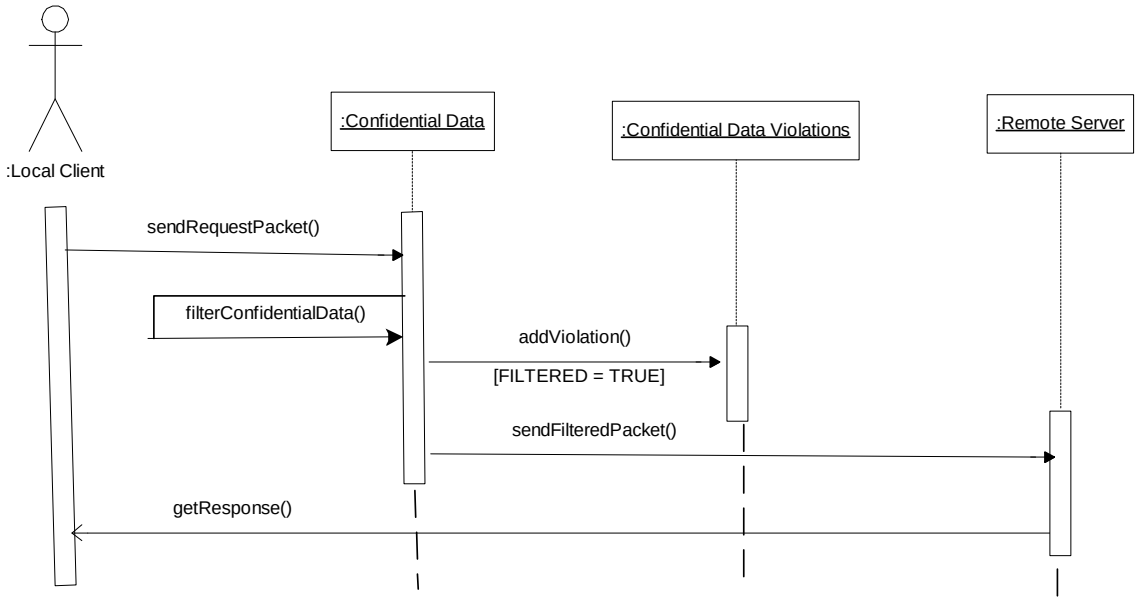
Sequence of Events for Applying Content Filtering (1st Algorithm)	
Main Sequence	1- Packets coming from Network Traffic Monitoring Module will be inspected to evaluate the Maliciousness Probability of the words in the packets. The maliciousOccurences and harmless Occurences of the words will be retrieved from the BayesWord table. 2- If the Maliciousness Probability of the whole packet is higher than the threshold value, the packet will be blocked. 3- The source IP, destination IP, time and the malicious content of the packet will be written to the FilteredContent table in the database.
Alternative Sequence	2 - If the Maliciousness Probability of the packet is lower than the threshold value, the packet will be allowed to pass.

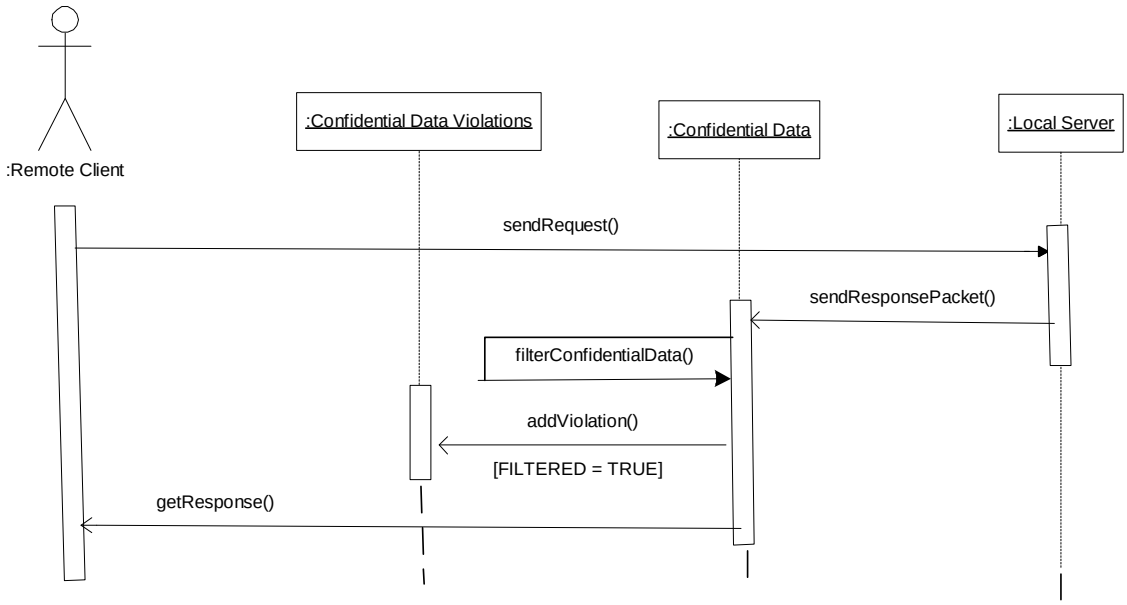
3.3.3.2 Applying Content Filtering (2nd Algorithm)



Sequence of Events for Applying Content Filtering (2nd Algorithm):	
Main Sequence	1- Packets coming from Network Traffic Monitoring Module will be inspected to evaluate the occurrences of black and white words of the user. (Users black and white words will be retrieved by making use of the UserGroups, BlackWordGroup, BlackWordList, ExtraWordGroup, ExtraWord and WhiteWordList table.) 2- The Maliciousness Probability of the packets will be computed. 3- If the Maliciousness Probability of the whole packet is higher than the threshold value, the packet will be blocked. 4- The source IP, destination IP, time and the malicious content of the packet will be written to the FilteredContent table in the database.
Alternative Sequence	3- If the Maliciousness Probability of the packet is lower than the threshold value, the packet will be allowed to pass.

3.3.3.3 Applying Confidential Data Filtering

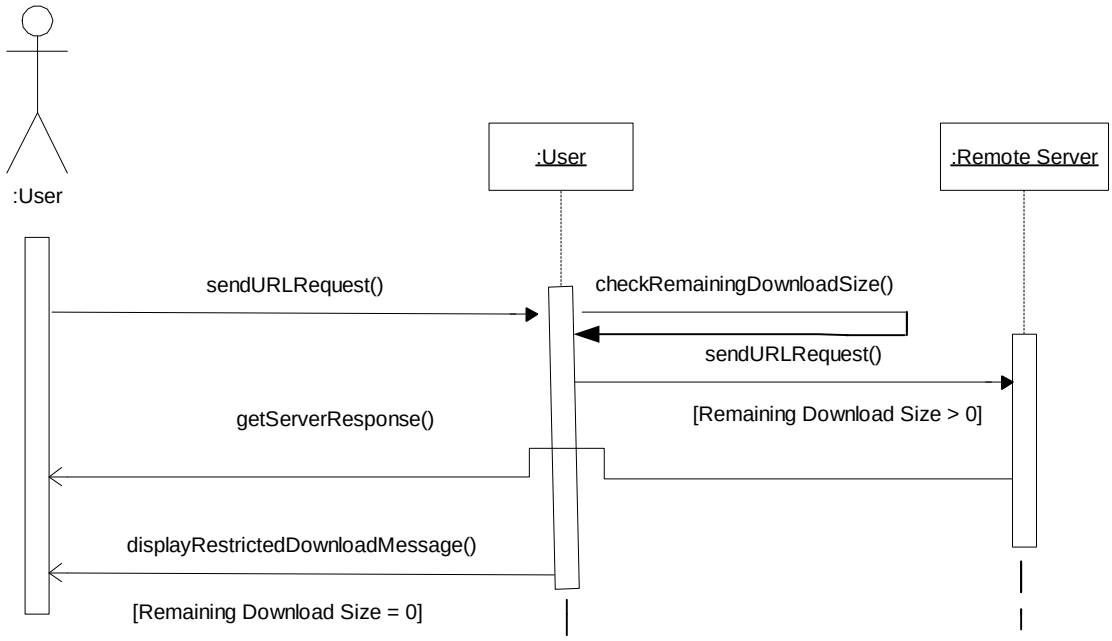




Sequence of Events for Applying Confidential Data Filtering	
Main Sequence	1- Packets coming from Network Traffic Monitoring Module will be inspected to find the occurrences of the confidential data, which is retrieved from ConfidentialData table. 2- If confidential data exists, the criticality of the packet will be evaluated. 3- If the criticality is higher than the threshold value, the packet will be blocked. 4- The source IP, destination IP, time and the confidential data in the packet will be written to the ConfidentialDataViolations table in the database.
Alternative Sequence	2- If no confidential data exists, the packet will be allowed to pass.
Alternative Sequence	3- If the criticality of the packet is lower than the threshold value, the packet will be allowed to pass.

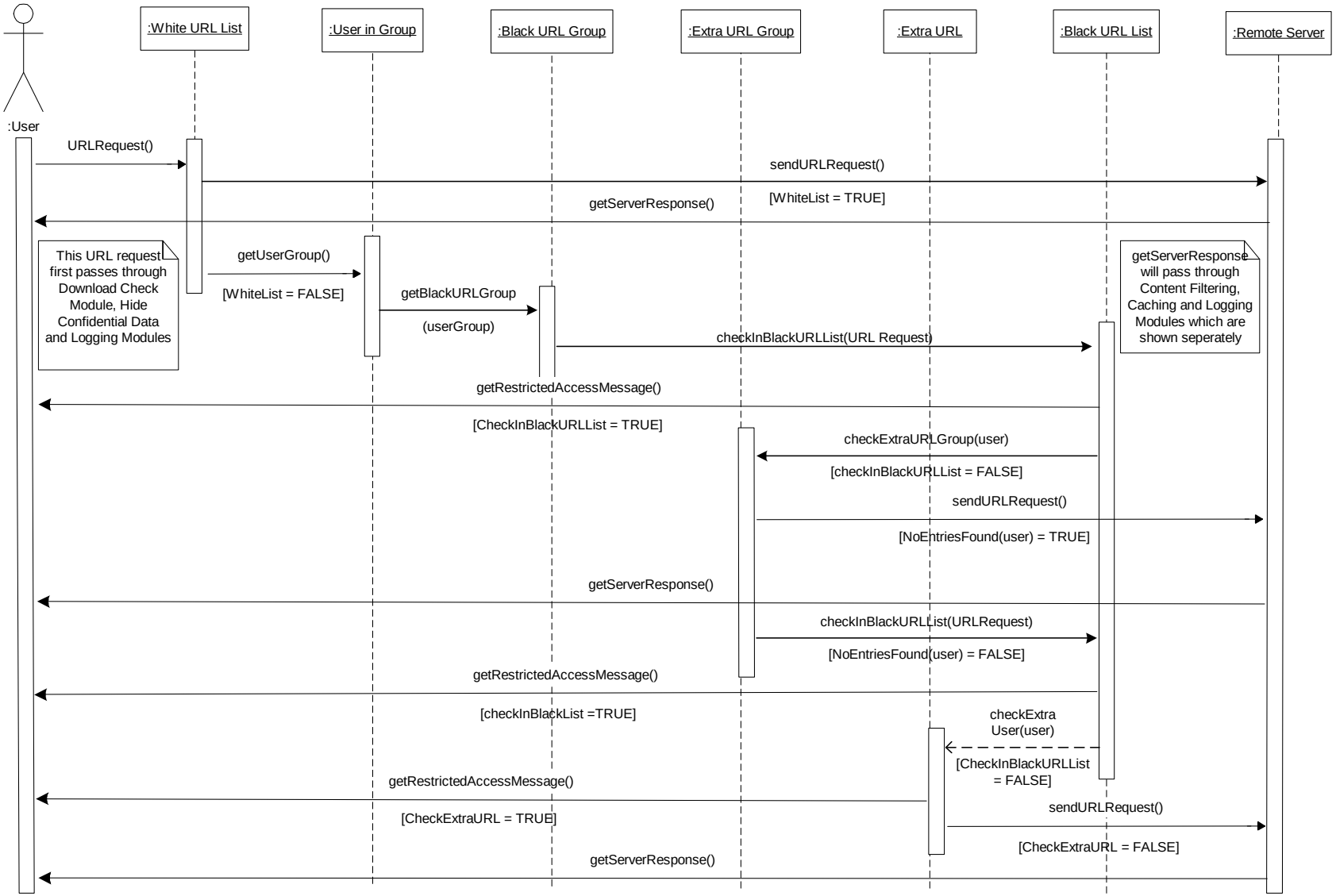
3.3.4 Restriction Module

3.3.4.1 Applying Download Restriction



Sequence of Events for Applying Download Restriction:	
Main Sequence	1- User sends a request to access a URL. 2- User's remaining download size will be checked from the User table in the database. 3- If the user has remaining download size which is greater than 0, he/she will be able to get the response for his/her request.
Alternative Sequence	3- If the user has exceeded his/her permitted download size, access will be rejected.

3.3.4.2 Applying URL Access Restriction

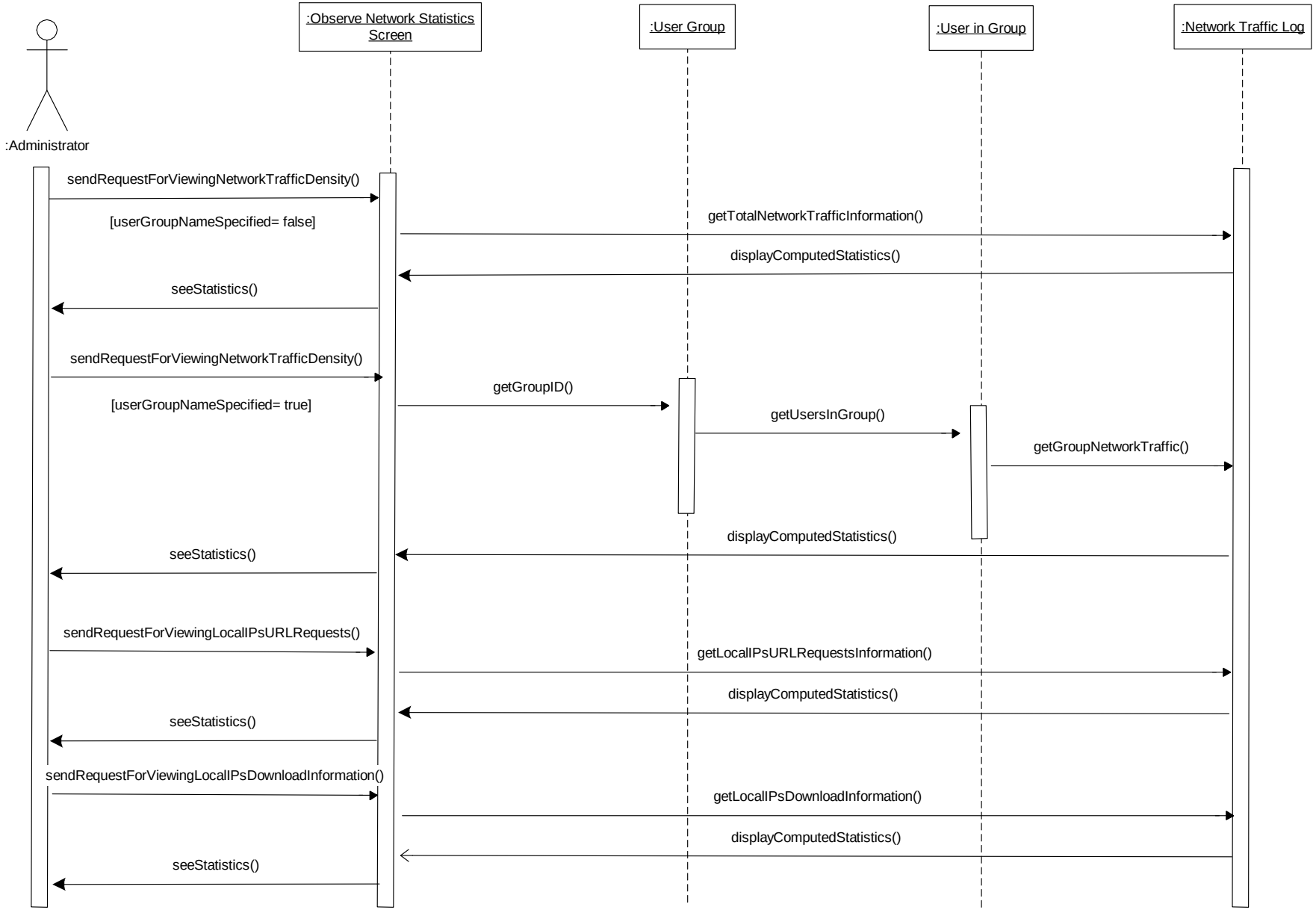


Sequence of Events for Applying URL Access Restriction:	
Main Sequence	1- User sends a request to access a URL.. 2- The URL is searched in the WhiteURLList table. 3- If the URL is not in the WhiteURLList, then the group of the user will be retrieved from the UserInGroup table. 4- System time is acquired to check against the time based restriction specified for each URL 5- According to the group of the user, BlackURLGroup which are restricted to the user will be retrieved. The URL is checked if it is in a restricted group from URLInGroup table and the restriction is valid at the moment. 6- If the URL is not in one of the restricted groups, then URL is checked against the ExtraURLGroups specified for the user and the restriction is valid at the moment. 7- If the URL is not in one of the ExtraURLGroups, the URL is checked against ExtraURL specified for the user and the restriction are valid at the moment. 8- If the URL is not restricted, then the packet will be forwarded to the destination.
Alternative Sequence	6- If the URL is in one of the restricted BlackURLGroup and the restriction is valid, then the packet will not be allowed to the network. 7- An error message will be displayed.

Alternative Sequence	8- If the URL is in the ExtraURL and the restriction is valid, then the packet will not be allowed to the network. 9- An error message will be displayed.
Alternative Sequence	4- If the URL is in the WhiteURLList table, the packet is allowed to pass to the network

3.3.5 Statistics Module

3.3.5.1 Computing Network Traffic, URL Request and Download Size Statistics

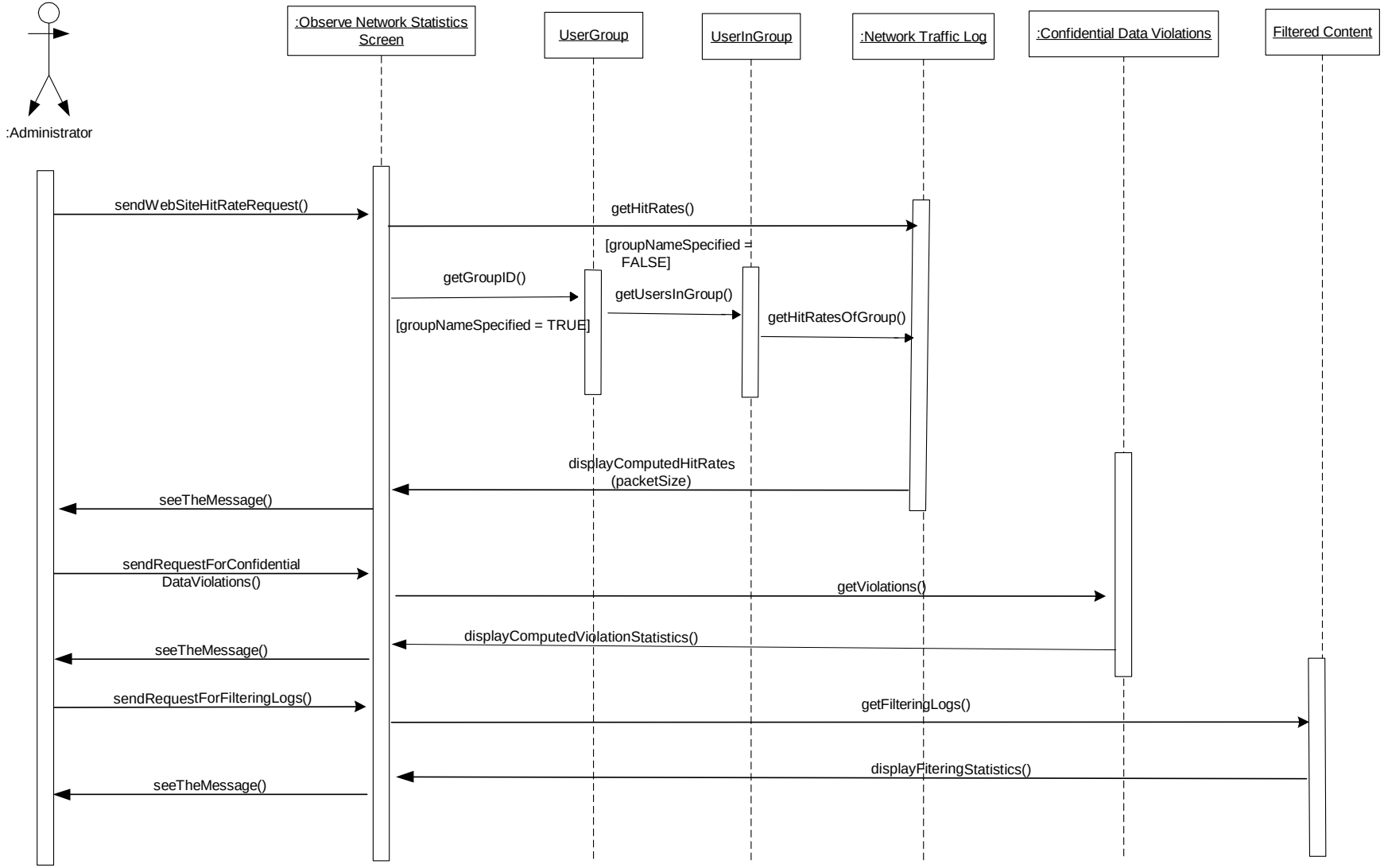


Sequence of Events for Viewing Network Traffic Density :	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to view the network traffic density through the web interface. 3- The statistics of the current day is computed and returned to the web interface.
Alternative Sequence	4- The administrator can choose a user group to view this group's statistics only. 5- Selected group's statistics will be returned to the web interface. 6- The statistics of the last month will be kept in the database. So, the administrator can specify a time interval to see the history statistics. 7- According to the specified time interval, the statistics will be computed and returned.

Sequence of Events for Viewing Local IP's URL Requests :	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to view a local IP's URL request statistics through the Web interface, by providing the local IP and time interval (hour, day, week). 3- The requested URL's, communication sizes and exact time information will be computed and returned to the Web interface.

Sequence of Events for Viewing Local IP's Download Information :	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to view a local IP's download information through the Web interface, by providing the local IP and time interval (hour, day, week). 3- If the user has exceeded the specified download limit, the time information will be computed and returned to the Web interface.

3.3.5.2 Computing Web Site Hit Rates, Confidential Data Violations and Filtering Logs



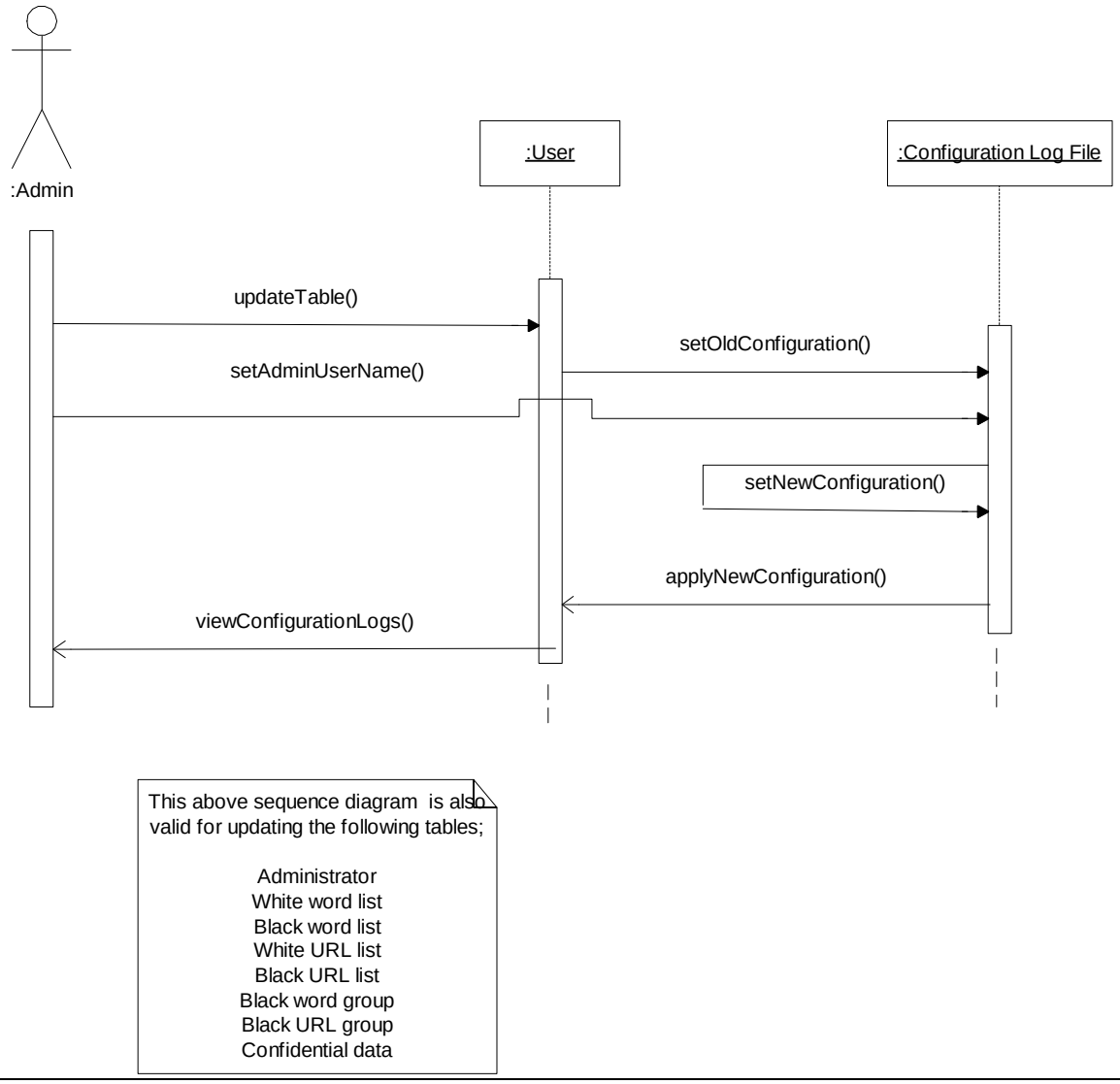
Sequence of Events for Viewing Hit Rates of Web Sites:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to view the hit rates of the most frequently accessed web sites through the Web interface.
Alternative Sequence	3- The hit rates are computed and returned to the Web interface. 4- The administrator can choose a user group to view this group's hit rates only. 5- Selected group's hit rates will be computed and returned.

Sequence of Events for Viewing Confidential Data Violations:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator sends a request to view the violations of confidential data protection, through the Web interface. 3- A list of violations ordered according to time will be returned to the Web interface.
Alternative Sequence	4- The administrator can choose a certain IP and time interval. 5- Selected IP's violated confidential data and exact time of violation will be returned to the Web interface.

Sequence of Events for Viewing Filtered Content Information:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- A list of filtered packets ordered according to time will be displayed.
Alternative Sequence	3- The administrator can choose a certain IP and time interval. 4- Selected IP's filtered content and exact time of request will be displayed on the screen.

3.3.6 Logging Module

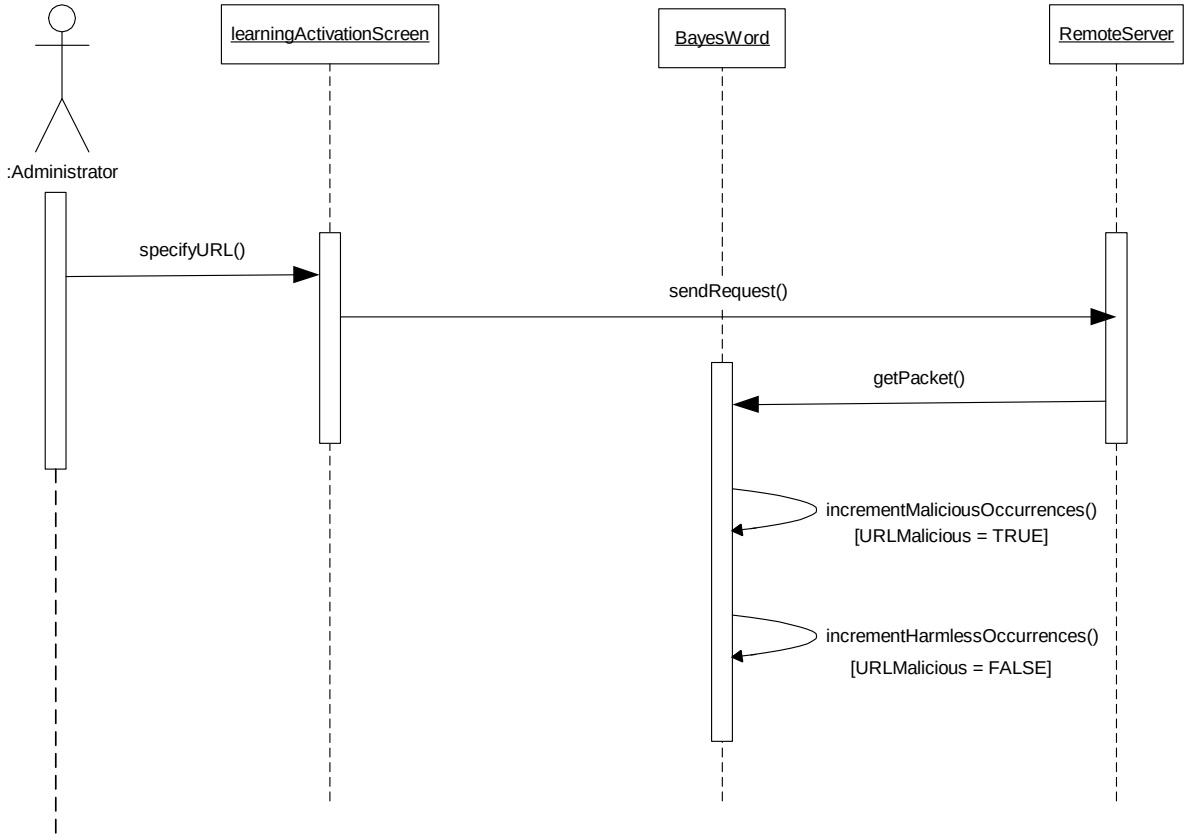
3.3.6.1 Saving the Configuration Logs



Sequence of Events for Saving the Configuration Logs:	
Main Sequence	1- Administrator sends a request to update a database table. 2- The old configuration existing in the table under consideration will be written to the ConfigurationLogFile. 3- Updates will be reflected to the database tables. 4- The new configuration will be written to the ConfigurationLogFile. 5- Administrator can view the configuration logs.

3.3.7 Learning Module

3.3.7.1 Learning

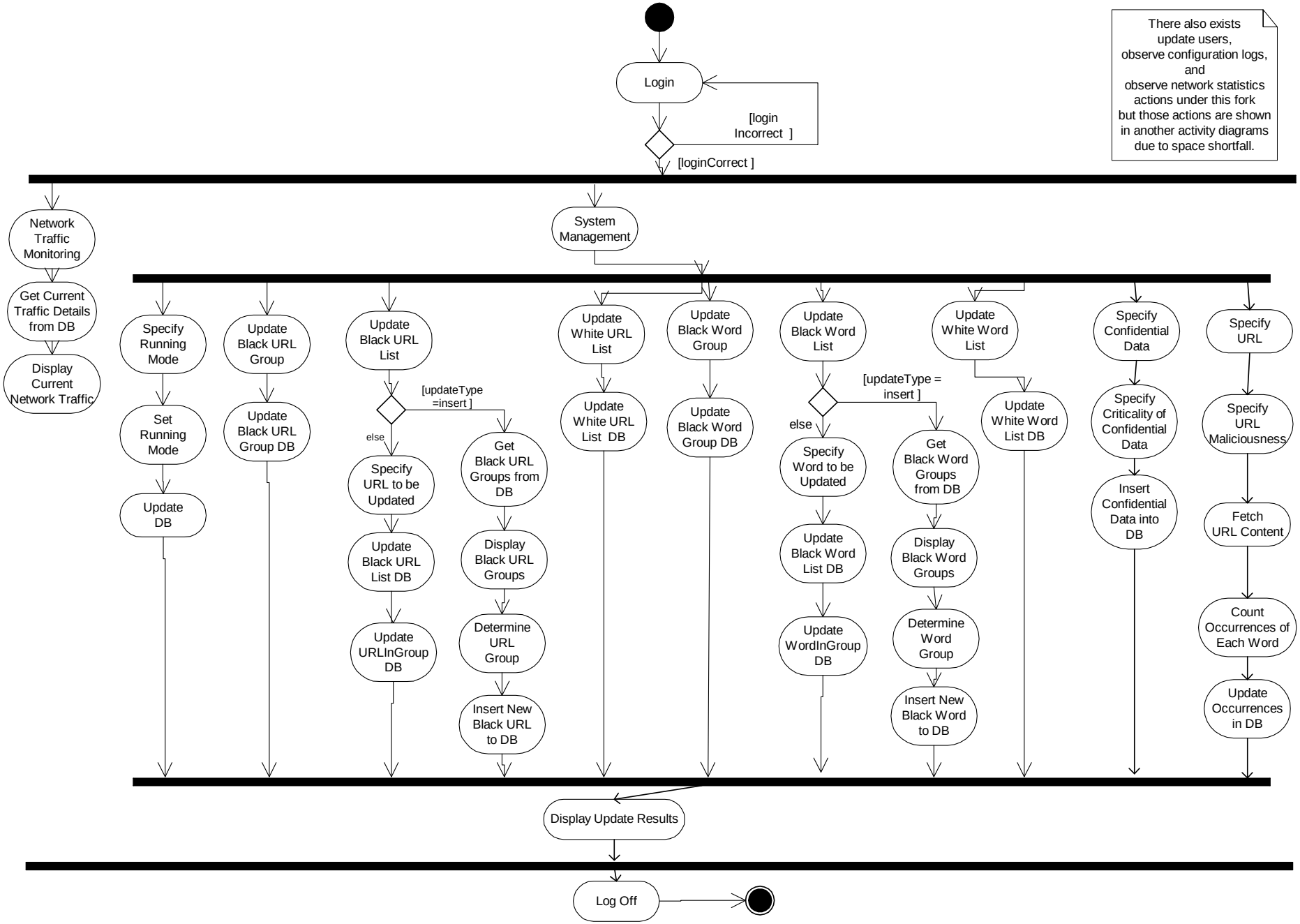


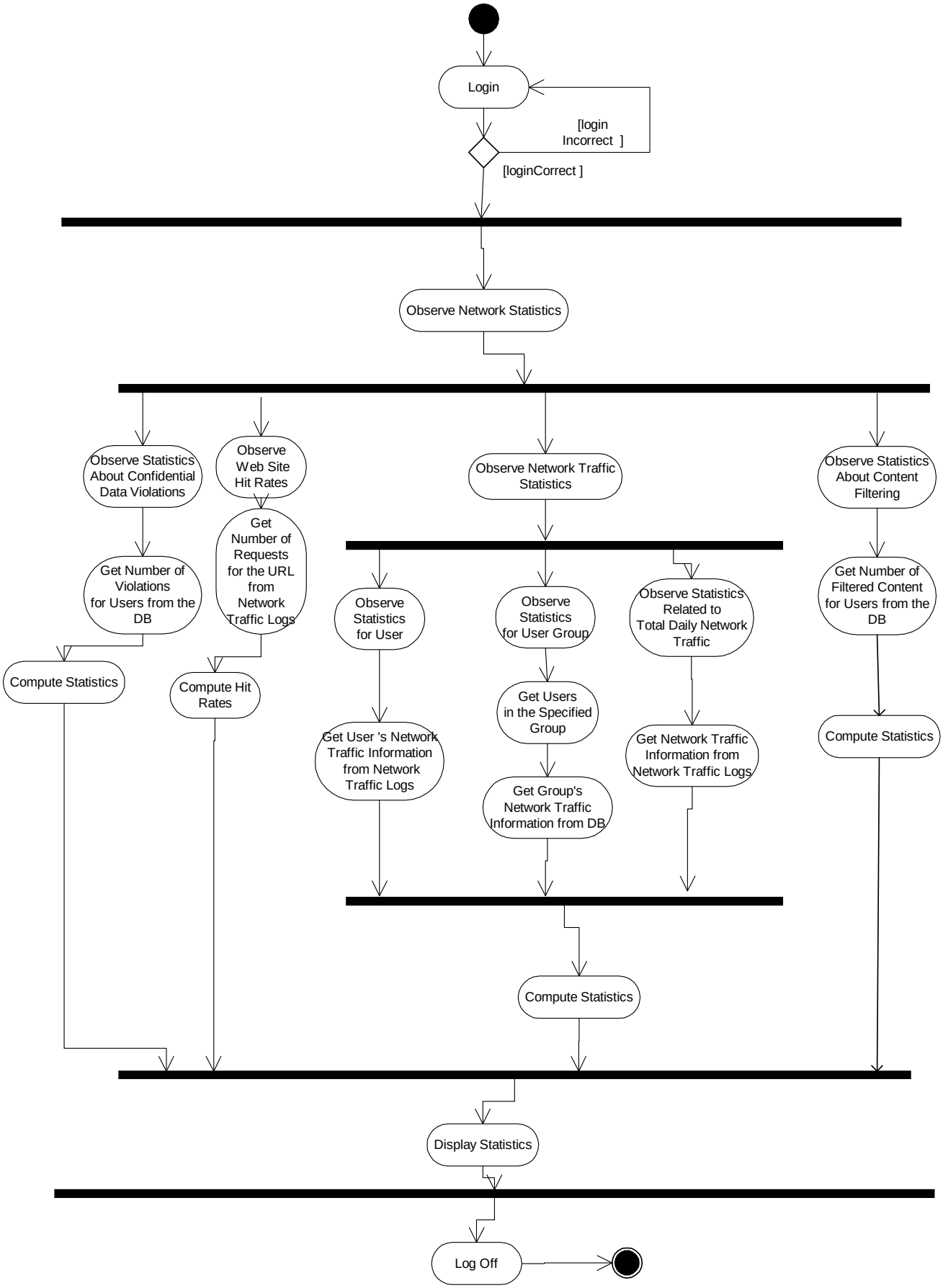
Sequence of Events for Learning:	
Main Sequence	1- Administrator must have logged in to the system and be granted the necessary permissions. 2- Administrator specifies the URL to be used for learning. 3- The URL request is sent to the remote server. 4- Packets returning from the remote server are inspected to count the occurrences of words. 5- If the URL is defined to be malicious by the administrator, malicious occurrences of each word are updated.
Alternative Sequence	5- If the URL is defined to be harmless by the administrator, harmless occurrences of each word is updated.

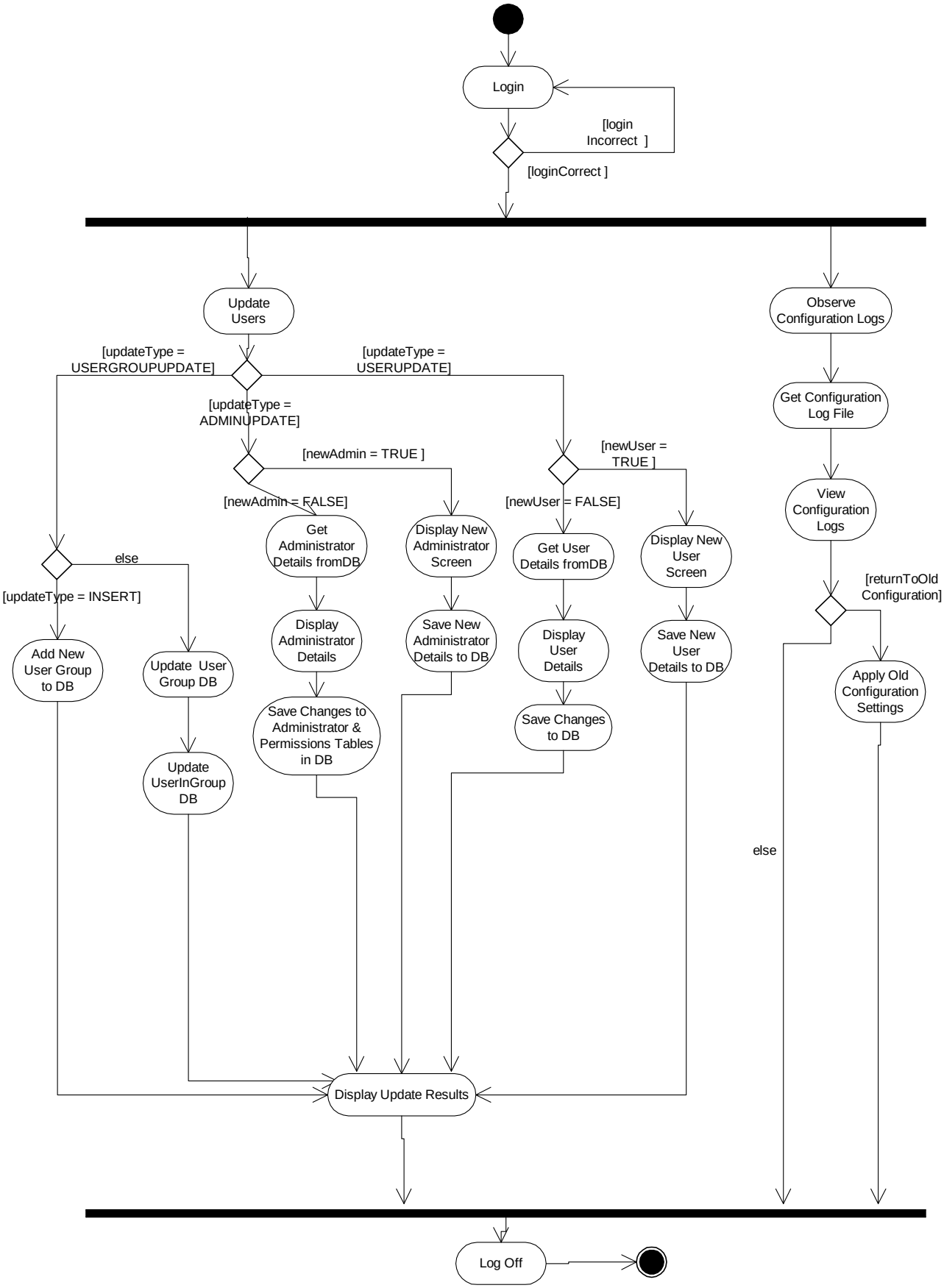
3.4 Activity Diagrams

3.4.1 System Management Module

Activity diagram of the System Management Module shows the flow of all controls that are applied by the administrator via the web interface. The web interface display functions are held by the System Management Module and for those displays the module has to interact with network traffic monitoring, statistics, and logging modules. These modules are integrated into the System Management Module in the activity diagrams for the better understandability of the system.

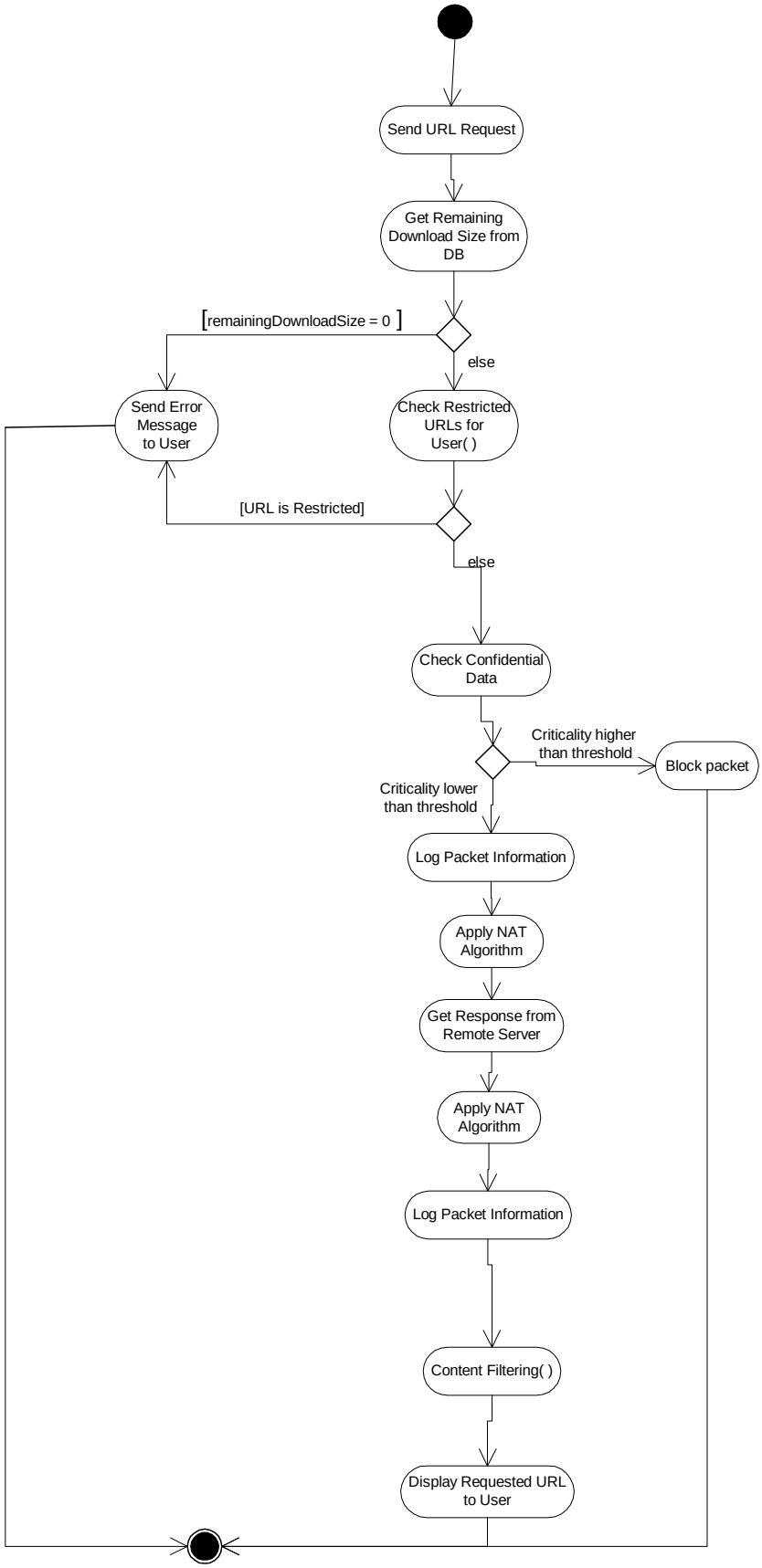

Login
[login Incorrect]
[loginCorrect]
There also exists update users, observe configuration logs, and observe network statistics actions under this fork but those actions are shown in another activity diagrams due to space shortfall.
Network Traffic Monitoring
Get Current Traffic Details from DB
Display Current Network Traffic
System Management
Specify Running Mode
Set Running Mode
Update DB
Update Black URL Group
Update Black URL Group DB
Update Black URL List
[updateType =insert]
else
Specify URL to be Updated
Update Black URL List DB
Update URLInGroup DB
Get Black URL Groups from DB
Display Black URL Groups
Determine URL Group
Insert New Black URL to DB
Update White URL List
Update White URL List DB
Update Black Word Group
Update Black Word Group DB
Update Black Word List
[updateType = insert]
else
Specify Word to be Updated
Update Black Word List DB
Update WordInGroup DB
Get Black Word Groups from DB
Display Black Word Groups
Determine Word Group
Insert New Black Word to DB
Update White Word List
Update White Word List DB
Specify Confidential Data
Specify Criticality of Confidential Data
Insert Confidential Data into DB
Specify URL
Specify URL Maliciousness
Fetch URL Content
Count Occurrences of Each Word
Update Occurrences in DB
Display Update Results
Log Off

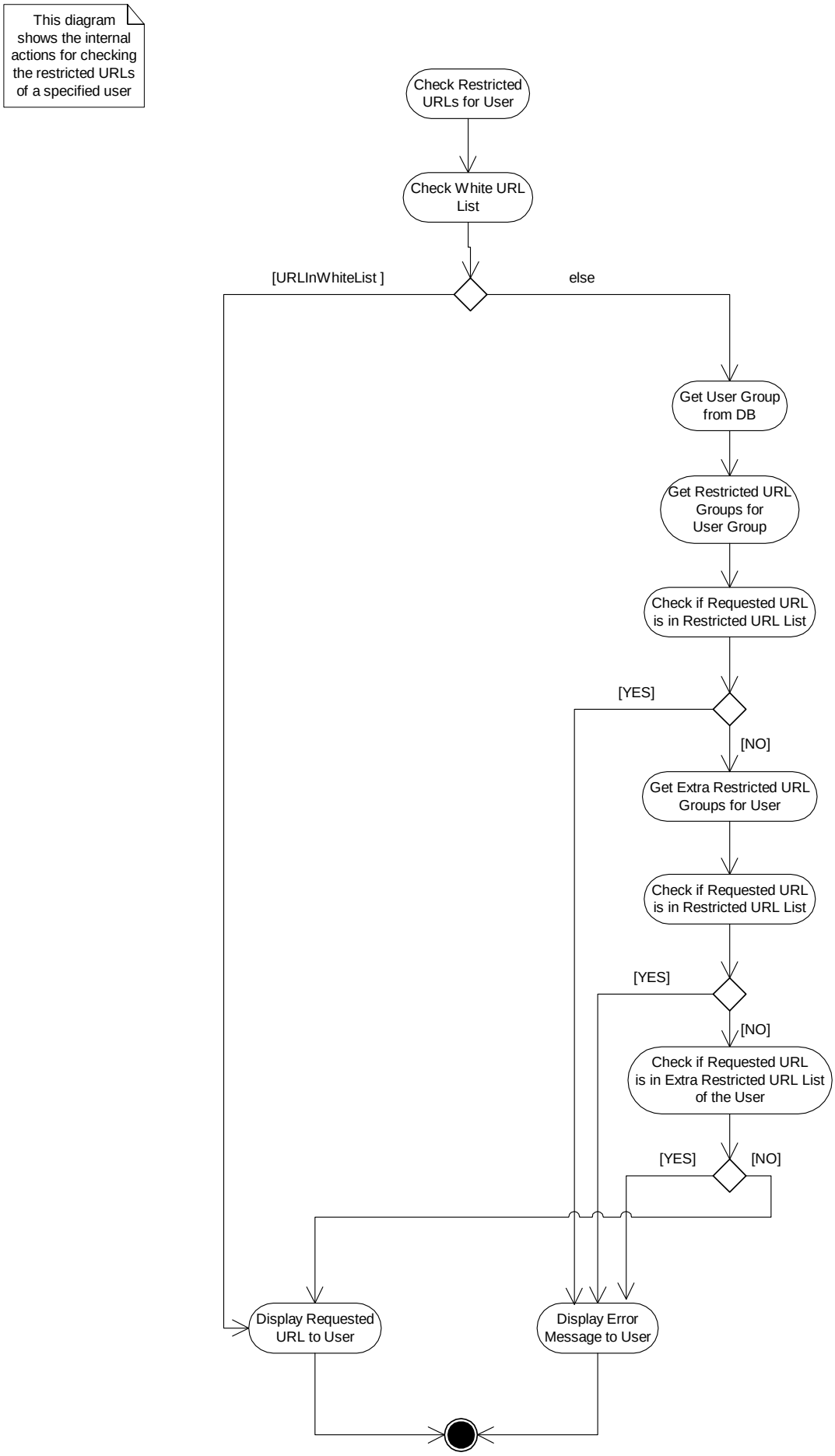

Login
[login Incorrect]
[loginCorrect]
Observe Network Statistics
Observe Statistics About Confidential Data Violations
Observe Web Site Hit Rates
Observe Network Traffic Statistics
Observe Statistics About Content Filtering
Get Number of Violations for Users from the DB
Get Number of Requests for the URL from Network Traffic Logs
Observe Statistics for User
Observe Statistics for User Group
Observe Statistics Related to Total Daily Network Traffic
Get Number of Filtered Content for Users from the DB
Compute Statistics
Compute Hit Rates
Get User 's Network Traffic Information from Network Traffic Logs
Get Users in the Specified Group
Get Network Traffic Information from Network Traffic Logs
Compute Statistics
Get Group's Network Traffic Information from DB
Compute Statistics
Display Statistics
Log Off

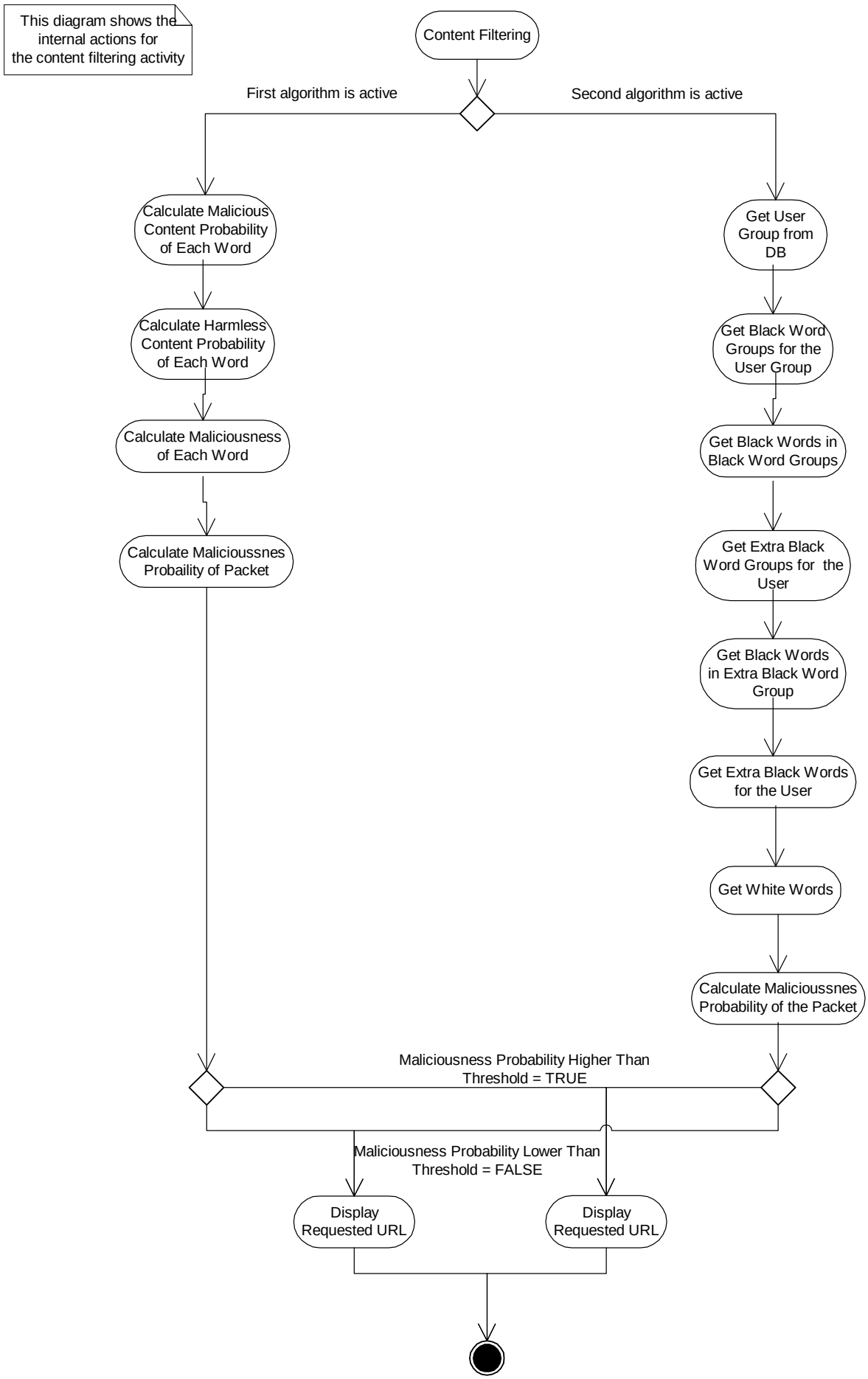

Login
[login Incorrect]
[loginCorrect]
Update Users
Observe Configuration Logs
[updateType = USERGROUPUPDATE]
[updateType = USERUPDATE]
[updateType = ADMINUPDATE]
Get Configuration Log File
[newAdmin = TRUE]
[newUser = TRUE]
View Configuration Logs
[newAdmin = FALSE]
[newUser = FALSE]
Get Administrator Details fromDB
Display New Administrator Screen
Get User Details fromDB
Display New User Screen
else
[updateType = INSERT]
[returnToOld Configuration]
Display Administrator Details
Save New Administrator Details to DB
Display User Details
Save New User Details to DB
Add New User Group to DB
Update User Group DB
Save Changes to Administrator & Permissions Tables in DB
Save Changes to DB
Apply Old Configuration Settings
Update UserInGroup DB
else
Display Update Results
Log Off

3.4.2 Restriction, Content Filtering, and Logging Modules

The activity diagram for restriction, content filtering, and logging modules are also integrated again for the better understandability issues.




This diagram shows the internal actions for checking the restricted URLs of a specified user
Check Restricted URLs for User
Check White URL List
[URLInWhiteList]
else
Get User Group from DB
Get Restricted URL Groups for User Group
Check if Requested URL is in Restricted URL List
[YES]
[NO]
Get Extra Restricted URL Groups for User
Check if Requested URL is in Restricted URL List
[YES]
[NO]
Check if Requested URL is in Extra Restricted URL List of the User
[YES]
[NO]
Display Requested URL to User
Display Error Message to User


This diagram shows the internal actions for the content filtering activity
Content Filtering
First algorithm is active
Second algorithm is active
Calculate Malicious Content Probability of Each Word
Calculate Harmless Content Probability of Each Word
Calculate Maliciousness of Each Word
Calculate Maliciousness Probaility of Packet
Get User Group from DB
Get Black Word Groups for the User Group
Get Black Words in Black Word Groups
Get Extra Black Word Groups for the User
Get Black Words in Extra Black Word Group
Get Extra Black Words for the User
Get White Words
Calculate Maliciousness Probability of the Packet
Maliciousness Probability Higher Than Threshold = TRUE
Maliciousness Probability Lower Than Threshold = FALSE
Display Requested URL
Display Requested URL

4 DATABASE DESIGN

Construction of the database is the key concept in the development of our project. This is because all other modules of our system are dependent on the data modeling. If the database design is made in a complete manner, the rest of the system will also be designed concretely.

4.1 Database Table Specifications

The following table specifications explain our project's database tables in a detailed manner.

4.1.1 LocalUser

Name	Content Description	Supplementary Info.
IP	Char(15)	Primary Key
name	VARCHAR(30)	Not Null
permitted Download Size	Float	
remaining Download Size	Float	

4.1.2 NetworkTrafficLog

Name	Content Description	Supplementary Info.
communicationID	Serial	Primary Key
sourceIP	Char(15)	Not Null
destinationIP	Char(15)	Not Null
destinationURL	VARCHAR(150)	
packetSize	Float	
time	Timestamp	

4.1.3 BlackWordList

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
word	VARCHAR(20)	Not Null
isActive	boolean	

4.1.4 BlackWordGroup

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
wordGroup	VARCHAR(20)	Not Null
isActive	boolean	

4.1.5 WhiteWordList

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
word	VARCHAR(20)	
isActive	boolean	

4.1.6 WordInGroup

Name	Content Description	Supplementary Info.
	Integer	Foreign Key (BlackWordList(ID))
wordGroupID	Integer	Foreign Key (BlackWordGroup(ID))

4.1.7 BayesWord

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
word	VARCHAR(20)	Not Null
maliciousOccurrences	int	
harmlessOccurrences	int	
isActive	boolean	

4.1.8 ConfidentialData

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
data	VARCHAR(30)	
criticality	integer	Not Null
isActive	boolean	

4.1.9 BlackURLList

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
URL	VARCHAR(150)	Not Null
isActive	boolean	
timeInterval	VARCHAR(30)	

4.1.10 BlackURLGroup

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
URLGroup	VARCHAR(20)	Not Null
isActive		

4.1.11 URLInGroup

Name	Content Description	Supplementary Info.
URLID	Integer	Foreign Key (BlackURLList(ID))
URLGroupID	Integer	Foreign Key (BlackURLGoup(ID))

4.1.12 WhiteURLList

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
URL	VARCHAR(150)	
isActive	boolean	

4.1.13 Administrator

Name	Content Description	Supplementary Info.
username	VARCHAR(30)	Primary Key
password	VARCHAR(20)	
IP	Char(15)	
fullName	VARCHAR(30)	
email	VARCHAR(30)	
GSM	Char(11)	

4.1.14 Permissions

Name	Content Description	Supplementary Info.
ID	Integer	Primary Key
type	Char(20)	

4.1.15 HavePermissions

Name	Content Description	Supplementary Info.
username	Integer	Foreign Key (Administrator(username))
permissionID	Char(20)	Foreign Key (Permissions(ID))

4.1.16 LocalUserGroup

Name	Content Description	Supplementary Info.
ID	Serial	Primary Key
groupName	VARCHAR(20)	
permittedDownloadSize	Float	

4.1.17 RestrictedURLforUserGroup

Name	Content Description	Supplementary Info.
userGroupID	Integer	Foreign Key (LocalUserGroup(ID))
blackURLGroupID	Integer	Foreign Key (BlackURLGroup(ID))

4.1.18 RestrictedWordforUserGroup

Name	Content Description	Supplementary Info.
userGroupID	Integer	Foreign Key (LocalUserGroup(ID))
blackWordGroupID	Integer	Foreign Key (BlackWordGroup(ID))

4.1.19 UserInGroup

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
userGroupID	Integer	Foreign Key (LocalUserGroup(ID))

4.1.20 ExtraURL

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
blackURLID	Integer	Foreign Key (BlackURLList(ID))

4.1.21 ExtraWord

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
blackWordID	Integer	Foreign Key (BlackWordList(ID))

4.1.22 ExtraURLGroup

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
blackURLGroupID	Integer	Foreign Key (BlackURLGroup(ID))

4.1.23 ExtraWordGroup

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
blackWordGroupID	Integer	Foreign Key (BlackWordGroup(ID))

4.1.24 ConfidentialDataViolations

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
violatedRuleID	Serial	Foreign Key (ConfidentialData(ID))
time	Timestamp	

4.1.25 RunningMode

Name	Content Description	Supplementary Info.
modeID	Integer	Primary Key
isActive	boolean	

4.1.26 FilteredContent

Name	Content Description	Supplementary Info.
userIP	Char(15)	Foreign Key (LocalUser(IP))
filteredWordID	Serial	Foreign Key (BlackWord(ID))
filteredBayesID	Serial	Foreign Key (BayesWord(ID))
time	Timestamp	

4.2 Database Table SQL's

After the table specifications, we have also constructed the database of our project in PostgreSQL with the following sql create commands:

```
create LocalUser(
        IP char(15),
        name varchar(30) not null,
        permittedDownloadSize float,
        remainingDownloadSize float,
        primary key(IP)
);

create table NetworkTrafficLog(
        communicationID serial primary key,
        sourceIP char(15) not null,
        destinationIP char(15) not null,
        destinationURL varchar(150),
        packetSize float,
        time timestamp
);

create table BlackWordList (
        ID serial primary key,
        word varchar(20) not null,
        isActive boolean
);

create table BlackWordGroup (
        ID serial primary key,
        wordGroup varchar(20) not null,
        isActive boolean
);

create table WhiteWordList (
        ID serial primary key,
        word varchar(20) not null,
        isActive boolean
);

create table WordInGroup (
        wordID integer,
        wordGroupID integer,
        primary key(wordID, wordGroupID),
        foreign key (wordID) references blackWordList on delete cascade,
        foreign key (wordGroupID) references blackWordGroup on delete cascade
);
```

```
create table BayesWord(
        ID serial primary key,
        word varchar(20) not null,
        maliciousOccurences integer,
        harmlessOccurences integer,
        isActive boolean
);

create table ConfidentialData (
        ID serial primary key,
        data varchar(30)
        criticality integer not null,
        isActive boolean
);

create table BlackURLList (
        ID serial primary key,
        URL varchar(150) not null,
        timeInterval  varchar(30),
        isActive boolean
);

create table BlackURLGroup (
        ID serial primary key,
        URLGroup varchar(20) not null,
        isActive boolean
);

create table URLInGroup (
        URLID integer,
        URLGroupID integer,
        primary key(URLID, URLGroupID),
        foreign key (URLID) references blackURLList on delete cascade,
        foreign key (URLGroupID) references blackURLGroup on delete cascade
);

create table WhiteURLList (
        ID serial primary key,
        URL varchar(150) not null,
        isActive boolean
);

create table Administrator(
        userName varchar(30),
        password varchar(20),
        IP char(15),
        fullName varchar(30),
        email varchar(30),
        GSM char(11),
        primary key (userName));
```

```
create table Permissions (
        ID integer,
        type char(20),
        primary key (ID)
);


create table HavePermissions (
        userName varchar(30),
        permissionID integer,
        primary key (userName, permissionID) ,
        foreign key (userName) references administrator on delete cascade on
                update cascade,
        foreign key (permissionID) references permissions(ID) on delete
                cascade on update cascade
);

create table LocalUserGroup(
        ID serial primary key,
        groupName varchar(20),
        permittedDownloadSize float
);

create table RestrictedURLforUserGroup(
        userGroupID integer,
        blackURLGroupID integer,
        primary key(userGroupID,blackURLGroupID),
        foreign key(userGroupID) references LocalUserGroup(ID) on delete
                cascade,
        foreign key(blackURLGroupID) references blackURLGroup(ID) on delete
                cascade
);

create table RestrictedWordforUserGroup(
        userGroupID integer,
        blackWordGroupID integer,
        primary key(userGroupID,blackWordGroupID),
        foreign key(userGroupID) references LocalUserGroup(ID) on delete
                cascade,
        foreign key(blackWordGroupID) references blackWordGroup(ID) on delete
                cascade
);
```

```
create table UserInGroup(
        userIP char(15),
        userGroupID integer,
        primary key(userIP,userGroupID),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key(userGroupID) references LocalUserGroup(ID) on delete
                cascade
);

create table ExtraURL(
        userIP char(15),
        blackURLId integer,
        primary key(userIP,blackURLId),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key (blackURLId) references blackURLList(ID) on delete
                cascade
);

create table ExtraWord(
        userIP char(15),
        blackWordId integer,
        primary key(userIP,blackWordId),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key (blackWordId) references blackWordList(ID) on delete
                cascade
);

create table ExtraURLGroup(
        userIP char(15),
        blackURLGroupId integer,
        primary key(userIP,blackURLGroupId),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key (blackURLGroupId) references blackURLGroup(ID) on delete
                cascade
);

create table ExtraWordGroup(
        userIP char(15),
        blackWordGroupId integer,
        primary key(userIP,blackWordGroupId),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key (blackWordGroupId) references blackWordGroup(ID) on
                delete cascade);
```

```
create table ConfidentialDataViolations(
        userIP Char(15),
        violatedRuleID serial,
        time timestamp,
        primary key(userIP,violatedRuleID),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key (violatedRuleID) references confidentialData(ID) on
                delete cascade
);

create table RunningMode (
        ID integer,
        isActive boolean,
        primary key (ID)
);

create table FilteredContent(
        userIP Char(15),
        violatedWordID serial,
        violatedBayesID serial,
        time timestamp,
        primary key(userIP,violatedRuleID),
        foreign key (userIP) references LocalUser(IP) on delete cascade on
                update cascade,
        foreign key (violatedWordID) references BlackWord(ID) on
                delete cascade
        foreign key (violatedBayesID) references BayesWord(ID) on
                delete cascade
);
```

5 NAMING AND COMMENTING CONVENTIONS

As a part of standardization process for rapid development of our project, we have decided to develop special naming conventions. In addition, we have decided the syntax of descriptive comments that will be used for understandability and maintainability of our product. The details of the specifications are described below.

5.1 Naming Conventions

We have decided that all names should be comprehensible. For names that are composed of more than one word, lower case/upper case characters will be used to distinguish between consecutive words.

Naming the Classes:

All classes will have names beginning with a capital letter. The classes with more than one word will have a capital letter at the beginning of each word. For instance, “NetworkTrafficLog” is a valid class name for our project.

Naming the Class Attributes:

Attributes will begin with a lower case letter. In case there are more words, they will be distinguished by capital letters at the beginning. “groupName” and “word” are valid class attribute examples.

Naming the Class Methods:

Methods will have the same convention with the class attributes.

Naming the Database Table:

Names of the tables in the database will begin with capital letters and will continue with a capital letter for each consecutive word. Attributes of the tables will follow the naming convention for the class attributes; that is, will begin with lower case letters and continue with upper case letters for each new word.

Naming the Files:

Files that include the source code and header for a class will be named as the following respectively:

<class_name><.cpp>

<class_name><.h>

Naming the Global and Local Variables:

We will try to avoid using global variables as an appropriate software engineering principle. However, in case of any necessity, global variables will be prefixed with “g_”, since usage of global variables significantly decrease the understandability of the source code. Likewise, pointers will be prefixed by “p_”. Since we will implement our project in C++, we will need extensive use of pointers. For local variables, we have decided to use a convention that will help differentiate the type of the variable, such as “an_int” for a variable of integer type.

5.2 Commenting Conventions

In order to increase the understandability of our source code appropriate commenting is an important concern. We are intending to use comments for file descriptions, for function definitions, and for not easily understood variables. Commenting style for our project is described as follows;

Commenting the Files:

Files should be described at the beginning according to the following format;

```
/* ----------------------------------------------------------------------------------------
/* File name:
/* Created by:
/* Created at: ( Date:DD.MM.YY – Time: HH:MM:SS)
/* Modified by:
/* Modified at: ( Date:DD.MM.YY – Time: HH:MM:SS)
/* Version:
/* Description:
----------------------------------------------------------------------------------------*/
```

Commenting the Functions:

For the description of the functions we have specified the following format;

```
/*-----------------------------------------------------------------------------------
/* Function Signature: <return_type> <function_name> (<param_1>,<param_2>,..)
/* Parameters: <parameter_name> <parameter_description>
/* Return value: <return_value> <return_value_description>
/* Function Description:
-----------------------------------------------------------------------------------*/
```

Commenting the Variables:

At the point of variable declaration a brief description could be added as follows;

```
<variable_type><variable_name> // variable description
```

6 HARDWARE AND SOFTWARE SPECIFICATIONS

6.1 Software Specifications

The system should provide a Linux operating system with the following facilities for our program to run:

- Apache web server,
- PostgreSQL as the Database Management System,
- A firewall (Iptables),
- A web browser for the administrative purposes,
- GNU C++ compiler.

6.2 Hardware Specifications

The following hardware should be provided for our program to run appropriately:

- Minimum 512 MB RAM,
- Minimum 5 GB of free disk space, for database storage,
- A Pentium IV processor,
- Minimum two network interface cards.

6.3 Tool Specifications

In the implementation phase of our project, we have determined to use the following tools:

- Linux operating system as the development platform,
- C++ programming language,
- GNU C++ compiler,
- PostgreSQL Database Management System,
- PHP and Apache web server.

7 TESTING PROVISIONS

7.1 Testing Considerations

It is our vision that no software product with extensive modules, appealing graphics, or sophisticated feedback mechanisms can be appreciated, if it fails in its essentials. With this vision in mind, we have planned to undertake a wide range of tests, which will assure the quality of our software and let us realize the deficiencies in the implementation, the integration of different modules and the performance of the overall system. With these tests, we are aiming to discover most of the inevitable errors as soon as possible, and delivering an error-free product to the end user.

To clarify the procedure to be followed, we have decided to apply systematic tests in the following order:

- **White box / Black box testing of individual modules**

  We will apply white box testing to our modules as soon as their implementation is over. We are expecting that the implementation details will not have been forgotten, so testing will be more efficient. Also, we will have realized errors before they will propagate to the later phases of implementation, since they will require greater amount of configuration management afterwards. As our white box testing strategy, we will try to monitor the module during execution and check if the module runs as expected.

  Black box testing will be used as a complementary strategy since it is useful in discovering unexpected behavior rapidly. We will apply black box boundary tests and try to find out problematic cases.

- **The integration tests of module groups**

  Although the testing of modules individually is necessary, it is by no means sufficient, since modules can behave very unexpectedly when integrated. We will apply black box strategies to check the integration of modules, and compare and contrast our expectations with the outcomes.

- **The integration test of the overall system**

  The overall integration tests will be carried out when the implementation of all modules is over. This is the most general testing, and will be used to make sure that the overall system can be integrated as expected.

- **Performance tests of the overall system**

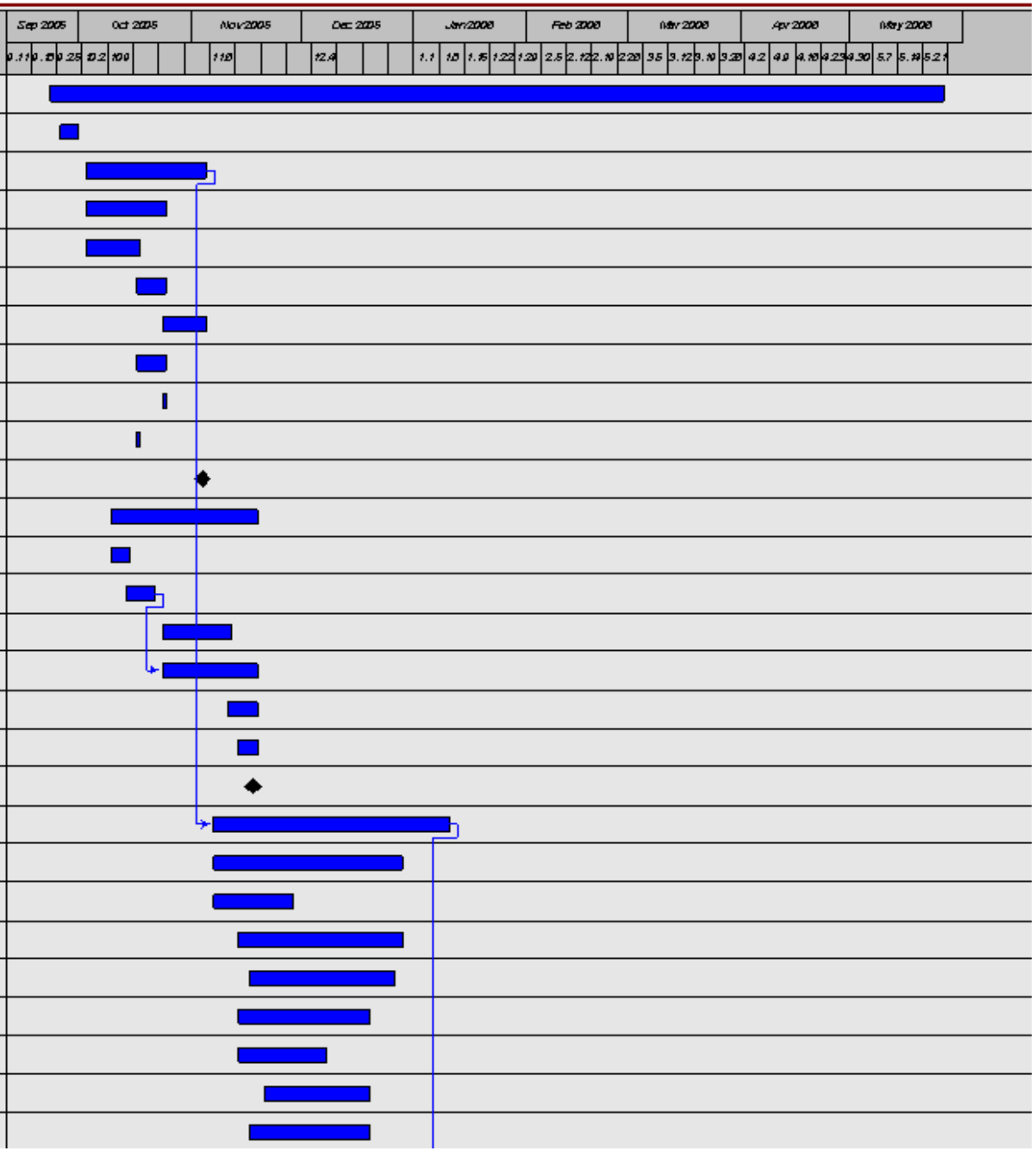
  Our system depends totally on performance issues to be usable. As a consequence, we will be holding a fair amount of performance tests to discover how the system will respond to the user needs. These tests will be carried both in large and small scale (that is, system-wise and module-wise) to find out if extra measures are needed to enhance the performance.

- **Stress tests of the overall system**

  Another issue is that our system has a high risk of running under too much work load, especially at times when the network traffic density is very high and content filtering tasks expected are complicated. As a result, we are aware that stress testing is crucial for our product. We will try to simulate high-stress scenarios to predict possible breakdowns of the product, and take the necessary precautions.

7 UPDATED GANNT CHART

ID	Task Name	Start	End
1	NetCheck Project	23.09.2005	26.05.2006
2	NetCheck Project Proposal	26.09.2005	30.09.2005
3	Analysis	03.10.2005	04.11.2005
4	Requirement Analysis	03.10.2005	24.10.2005
5	Literature Survey	03.10.2005	17.10.2005
6	Meeting with Customers	17.10.2005	24.10.2005
7	Analysis Report Writing	24.10.2005	04.11.2005
8	Risk Analysis	17.10.2005	24.10.2005
9	Project Scheduling and Tracking	24.10.2005	24.10.2005
10	Project Quality Plan	17.10.2005	17.10.2005
11	Milestone	04.11.2005	04.11.2005
12	Gathering Background Information	10.10.2005	18.11.2005
13	Network Architecture	10.10.2005	14.10.2005
14	TCP/IP Protocols	14.10.2005	21.10.2005
15	Network Security	24.10.2005	11.11.2005
16	Linux Networking	24.10.2005	18.11.2005
17	Web Programming	11.11.2005	18.11.2005
18	Data Mining	14.11.2005	18.11.2005
19	Milestone	18.11.2005	18.11.2005
20	Design	07.11.2005	10.01.2006
21	Filtering and Learning ModuleDesign	07.11.2005	28.12.2005
22	Network Traffic Monitoring Design	07.11.2005	28.11.2005
23	Blocking and Restriction Module Design	14.11.2005	28.12.2005
24	Logging Module Design	17.11.2005	26.12.2005
25	System Manegement Module Design	14.11.2005	19.12.2005
26	Web Page Interface Design	14.11.2005	07.12.2005
27	Statistics Interface Design	21.11.2005	19.12.2005
28	Database Design	17.11.2005	19.12.2005



29	Running Modes Design	21.11.2005	28.12.2005
30	Initial Design Report	18.11.2005	02.12.2005
31	Milestone	02.12.2005	02.12.2005
32	Final Design Report	26.12.2005	06.01.2006
33	Milestone	09.01.2006	09.01.2006
34	Prototype Development	05.12.2005	16.01.2006
35	Coding Prototype	21.12.2005	16.01.2006
36	Prototype Demo	17.01.2006	17.01.2006
37	Implementation	11.01.2006	05.05.2006
38	Filtering and Learning Module Implementation	09.02.2006	09.03.2006
39	Network Traffic Monitoring Module Implementation	24.01.2006	14.02.2006
40	Blocking and Restriction Module Implementation	09.02.2006	28.02.2006
41	Logging Module Implementation	02.02.2006	23.02.2006
42	System Management Module Implementation	09.03.2006	13.04.2006
43	Web Page Interface Implementation	14.04.2006	03.05.2006
44	Statistics Interface Implementation	21.04.2006	28.04.2006
45	Database Implementation	10.01.2006	31.01.2006
46	Running Modes Implementation	07.04.2006	03.05.2006
47	Milestone	24.04.2006	24.04.2006
48	Testing	07.03.2006	19.05.2006
49	Unit Testing	07.03.2006	04.05.2006
50	Integration Testing	20.04.2006	19.05.2006
51	Milestone	22.05.2006	22.05.2006
52	Project Finalization	12.05.2006	26.05.2006
53	Application Setup Development	19.05.2006	25.05.2006
54	User Manual Preparation	12.05.2006	19.05.2006
55	Milestone	26.05.2006	26.05.2006