**CHAPTER THREE**

**3.0 SYSTEM ANALYSIS, METHODOLOGY AND DESIGN**

**3.1 Analysis of the Existing System**

Analyze properly the current state of crime reporting through our various security agencies (The police, DSS, Civil defence, Army, Local vigilante etc)

**3.2 Limitation of the Existing System**

Describe the current challenges of how these crimes are been reported, that is, the non-web based form of crime reporting. This should be in alignment with your statement of problem

**3.3 Justification of the New System**

Describe the advantages of the proposed web-based crime reporting system which you will develop. State clearly how it is a better improvement over the previous methods of reporting and recording crimes.

**3.4 Methodology**

**3.5 Data Analysis**

Describe vividly how data flows from logging in as a user and also as an administrator, how data flow in capturing and storing crime reports. Describe also how data flows in retrieving the stored crime reports from the database, describe how data flows in creating users, updating their profiles and deleting them as the case maybe. Describe the users of this application in order to define your access control privileges, that is, the url is not made public, admin will create users and super admins will create admins, reports should also be searched and filtered using key criteria like Reporting agency (NPF, NSCDC, DSS, Military), mode of report (In person, telephone, physical observation, email etc), state, local government, community, type of crime reported (Rape, theft, banditry, kidnap, murder, armed robbery, assault, child abuse, human trafficking, drugs trafficking, destruction of property, internet fraud, financial fraud, etc), date range (from one specific date to another specific date).

**3.6 Result of Analysis**

Give an overview of the result of the analysis for every important action and process

3.7 System modeling

There are quite a number of Unified Modeling Language (UML) notations that are currently in use for developing computing models, these notations include: use case, activity diagram, block diagram, transition chart, object class diagram, sequence diagram and so on. In this thesis, we chose the sequence diagram notation. This is because it can clearly illustrate every single process and flow of information in the model with less complexity and more clarity. The figure 3.1 is a sequence diagram of the proposed model.

**Web application Database**

**Successful Trace**

**Successful IP registration**

**Valid login**

**Successful Report**

**User Successfully Added**

**Logout Successful**

**Login Verify Login**

**Access Granted Login Ok Main Menu**

**<<exception>>**

**Invalid login**

**IP Reg request Enter IP info**

**Register IP IP Info OK <<exception>>**

**Reg Unsuccessful Trace Request Enter Source IP**

**Trace Complete Trace Initiated <<exception>> unsuccessful Trace Incomplete**

**Report Request Prepare Report Show Report Report Complete**

**Add User Request Enter User Info New User Added New User Info OK**

**Logout**

**Logout Successful**

Figure 3.1: Sequence Diagram of Proposed System

The model of the proposed system simplifies how the system should function and all the possible operations that can be performed on the system. This model shows every bit of interaction between the user, the web-based application and the database that drives the application. The first action is the login, this requires the user to provide a login credential (username and password). This login credentials will be verified to check for its validity and existence in the database, if they exist and are valid, access will be granted and user will be redirected to the main menu page where they will be able to perform other operations. On successful login, users will be notified and also when incorrect login credentials are provided, an error message will be thrown up notifying users to enter correct username and/or password. On the main menu, there are IP registration, IP trace, view report, add user and logout options.

The IP registration option requires the user to register packet header information by keying in the values of the information in a form provided and the register button posts the form values into the database. Feedback message will be sent to the users as to whether registration was successful or not.

The IP trace option, provides a form where the source IP address can be entered and on clicking the trace button, tracing begins and when tracing is completes, a message will be popped up notifying user of a successful trace and that results have been stored in the database. The view report option, upon request, it displays the content of all the result of trace stored in the database.

The add user options allows users of higher level such as administrators and super administrators to create legitimate users of the system. This option requires only an administrative privilege. This option contains a form page where user information can be provided. On clicking the add user button, the values entered in the form will be stored in the database. Upon success of this operation, administrators will be notified via a message. The logout option simply terminates the session and redirects users to the login page.

**3.8. Architecture of the Proposed Design System**

The proposed system is an application that can effectively trace network attacks. It does not just stand alone in the network. It needs other network devices like the router, firewall and IDS to operate smoothly. The architecture shows the flow of data from the broad Internet into a network of interest otherwise seen as a target of evaluation. It describes how each component relates with one another in the network towards achieving an effective network attack detection and traceback. This data flow and relationship is shown in figure 3.2.

Figure 3.2: Architecture of the Proposed Designed System

The figure 3.2 shows the architecture of the network architecture that defines where the proposed system sits on the network and how and when it should interact on the network with other networking devices. Network packets come into the network from the internet via the router, the router does its job of routing by either locating the destination address on its network or it forwards it to the destined network. The firewalls now inspect the incoming packets using any of its methods and policy of ingress filtering, stateful packet inspection or other methods. Network attacks like computer worms, viruses, denial of service etc. can easily evade firewalls because they are stealth and can pass through legitimate channel. This is because firewalls do not inspect the contents of packets that come into the network via legitimate channels. The next line of defense is the IDS. The packets are then allowed to pass through the IDS. IDSes can inspect the content and signature of packets, some IDSes can even inspect packet behaviour in order to determine which one is malicious or not. At this point mostly, attacks are detected and an alert is raised. The IDS (specifically NIDS) is connected directly to an application server. On the application server, the connected monitor will display the packet header information of the detected malicious packet, this information include: activity, activity time, source IP, destination IP, time to live (ttl) and type of protocol (top). This information is enough to start a network forensic investigation. At this point, the proposed system sitting on the application server is now tasked with the responsibility of storing packet header information, tracing the source IP and generating reports based on the results of trace. The report generated provides meaningful information pertaining to the attacker, theses information include: source IP, destination IP (which is now the IP address of the origin of the malicious packet), the DNS name and even the country location. Once attacks have been detected and alert raised, the malicious packets are now lured into the DMZ where they can be contained, this DMZ prevents the attack from going straight into the LAN and causing havoc on host systems and end devices.

In terms of security, the proposed system is strategically sitting in the network, specifically on the application server. The first line of security is the firewall, since attacks that can be launched on the application externally comes into the network as packets possibly containing Trojans, rootkits, worms or other forms of attack that could prejudice the security of the application by stealing legitimate usernames and passwords. In the eventuality that the firewall is by-passed, the IDS is placed directly behind the firewall to analyse and act on the packets. It will detect malicious activities, generates an alert and instead of allowing such packets into the servers and the rest of the network, the packets will be guided to a DMZ area where they can be contained.

More so, there will be policies defining individuals in an organization that will be granted access to the application, these people are mainly IT staff with the qualified privilege and responsibility to conduct attack traceback. These policies can include disclosure of the application’s user resource locator (URL) address, user’s usernames and passwords as well as restricting access to the server, desktop or laptop system of legal users. In the event of an attacker who succeeds in gaining access to the usernames and password stored in the database, which is practically difficult, the passwords are hashed using secure hash algorithm (SHA 1). In trying to break or decode the password hashes, time is consumed thereby prolonging security. Policies that permit daily or weekly changing of usernames and passwords could be implemented to render attackers helpless, so that even when they spend time breaking the password hash they obtained, which theoretically it takes several weeks and even years, it will be useless because that password would have been changed.

In terms of the assurance that the source IP obtained is original, the IDS is responsible for that. This is because all packets transmitting across the internet have headers, in the headers there are field that indicates the source IP address, destination IP address etc. The IDS is designed to extract the packet header information, identify the activity of the malicious packet, date and time stamp etc. all these information will be displayed on the screen of the server connected to the IDS on the network. This makes the acquisition of the source IP address reliable and authentic, except for the case where the IP address extracted was spoofed.

**3.8.2 The Input Database Interface**

The input database interface is an environment on the proposed web-based traceback application that provides a means for inputting into the database specific malicious packet header information captured by the IDS. It is basically a form page, it has the following fields: activity, date and time stamp, source IP, destination IP, type of protocol (TOP), time to live (TTL), destination port, source port. It has a register button that automatically stores the information entered on the form into the database. The input database interface and the table data is shown in figure 3.3 and table 3.2.

**Input Database Interface**

* Packet header information
* Activity
* Activity time
* Source IP
* Destination IP
* Source port
* Destination port
* TTL
* TOP

Figure 3.3: Input database interface

Figure 3.3: The Input Database Interface

***Put your own data fields based on what you designed***

Table 3.2: Input Database Table Data

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Activity** | **Activity time** | **Source IP** | **Dest IP** | **Source port** | **Dest Port** | **TTL** | **TOP** |
| 1 | Port scan | 23:36:12 | 192.168.17.1 | 10.10.0.1 | 25 | 80 | 125 | UDP |
| 2 | Ping sweep | 20:18: 07 | 108.101.10.77 | 192.168.25.2 | 88 | 1024 | 75 | TCP |

**3.8.3 The Output Database Interface**

The output database interface is a mere display of the result of database query. The main output interface is the display of results of trace in an understandable form. The output database interface and the table data is shown in figure 3.4 and table 3.3.

**Output Database Interface**

* Source IP
* Destination IP
* Trace Time
* DNS
* Hops
* Country Location
* Date processed

Figure 3.4: Output Database Interface

Table 3.3: Output Database Table Data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Source IP** | **Dest IP** | **Trace time (Secs)** | **DNS** | **Hops** | **Country Location** | **Date processed** |
| 1 | 192.168.0.100 | 10.10.1.12 | 120 | Iphase | 15 | Abuja | 10-07-2012 |
| 2 | 192.168.16.1 | 172.168.17.2 | 117 | FUT | 25 | Minna | 12-07-2012 |
| 3 | 105.108.1.28 | 192.168.31.9 | 90 | VIC | 10 | London | 14-07-2012 |

**3.9 Top-Down Design and Software**

**3.9.1 The Main Menu**

The main menu is just like a welcome page that displays to the user the types of operation he/she can perform and how to perform them. The figure 3.6(a) is the main menu design.

Main Menu

View crimereport

Add crime report

Manage users

Dashboard

Figure 3.6(a): The Main Menu Design

**3.9.2 The Subsystem Menu**

The proposed system is a web-based application. It simply requires the application to be launched via a web browser such as internet explorer, Mozilla Firefox etc. the whole application is broken down into processes, these processes include: login, IP registration, IP traceback, view report and logout. The figure 3.6 is the subsystem menu design.

Activity

Activity time

Src IP

Dest IP

Src Port

Dest Port

TTL

TOP

Register

First Name

Last Name

Username

Password

Ulevel

Email

Phone No

Add User

Main Menu

View crimereport

Manage user

Add crime report

Dashboard

Figure 3.6(b): The Subsystem Menu Design

**3.9.2.1 Login**

This is the first process in the proposed system. It is specifically responsible for restricting access only to authorised users. This process requires users to enter their username and password, it has a login button, this button allows the login credentials (username and password) to be verified that they are legit before access will be granted. The verification process requires the application to check if this login credentials exists in the user login table in the database, if they exist, it checks if they both match on the same row, if they do, it considers it legitimate and grants access to the application by starting a session and redirecting the user to the main menu. In the case where one of these conditions is not met, it denies access and requires that user should enter correct username and password. The figure 3.7 shows a flowchart describing this process.

Connect to database

Is username and password in database?

Select user login table in database

Accept login credentials, start a session and redirect user to main menu

Does username and password match on the same row?

Enter username and password

No

Yes

No

Yes

Yes

Yes

Figure 3.7: Flowchart of Login Process

**3.9.2.2 IP Registration**

This process is responsible for storing malicious packet header information in the IP registration table in the database. These information include: activity, source IP, destination IP, date and time stamp etc. after entering the corresponding values into their respective form fields, the register button posts them into their respective columns in the IP registration table in the database on the fly. The figure 3.8 is a flowchart describing this process.

Connect to the database

Select the IP registration table

Enter into IP registration table the corresponding values of the defined fields

Enter packet header information

Figure 3.8: Flowchart of IP Registration

**3.9.2.3 IP Traceback**

This is the process that is responsible for the actual IP trace. It does this based on the ICMP traceback message received. At the destination network, once malicious packets are detected, it is registered on the web-based traceback application server via the interface provided by the application. A trace button is responsible for first checking the ICMP header on the packet and further checks for the ICMP traceback message source field, this is where all the ICMP traceback messages from all the routers the packet traversed is lodged. The next thing it does is to construct a path based on the number of ICMP traceback message in the ICMP header source address field as found on the destination network to the source network, at this point it sets a counter on the condition that the ICMP message is less than or equal to the total number of ICMP traceback messages in the ICMP header and that the ICMP message decrement by one. The next step is to check for the ICMP traceback message source address of the router directly connected to the destination router (direct link). At this point the total number of ICMP traceback message is set again to decrement by one. It then checks out of the remaining ICMP traceback message the source address of the router directly connected to the previously identified source address, it update the current number of ICMP traceback message left. It repeats this process until the total number of ICMP traceback message equals zero, which signifies that trace is complete. The figure 3.9 is a flowchart describing this process.

Check ICMP traceback messages in the ICMP header

Construct a path to the source based on ICMP traceback messages

Initiate counter, for ICMP message <= n; ICMP message decrement by 1, set ICMP message = n-1. (n is total number of ICMP messages)

Check for the ICMP message source address of the router directly connected to the network

Update current value of ICMP message

Check from the ICMP header the source address of the ICMP message directly connected to previously checked ICMP message

Update current value of ICMP messages in the ICMP header

Is ICMP message = 0?

No

Yes

Figure 3.9: Flowchart of ICMP Traceback

**3.9.2.4 View Report**

This is the process responsible for displaying the output of traceback stored in the database. This process requires the web-based application to connect to the database, select the report table and show all contents. The figure 3.10 is a flowchart describing this process.

Connect to database

Select report table

Show all content of report table

Figure 3.10: Flowchart of View Report Process

**3.9.2.5 Add User**

This is the process responsible for creating users that will be legible for creating users that will be legible to use the application. This privilege is exclusive only to a higher administrator. This process involves verification of the logged user, that is, checking to see that the user logged has a higher administrative access privilege. It establishes a connection to the database, selects the user login table, it checks the login credentials to see that they match with a higher administrative right, if they do, the web-based application redirects the user to a form page where they can create a user profile by entering the information on the form page and submitting them to the database. This requires the application connecting to the database, selecting the user login table and entering into their respective column field their corresponding values. The figure 3.11 is a flowchart describing this process.

Connect to database

Select user login table

Is login credentials in database?

Is login credentials same as higher administrator?

Redirect user to add user form page

Connect to database, select user login table and update with information obtained from form page

Enter username and password

Enter user information

No

Yes

No

Yes

Yes

Figure 3.11: Flowchart of Add User

**3.9.2.6 Logout**

This is the process responsible for destroying a session, in doing so the application set the session to be false, it is a complete reverse of the login process. The figure 3.12 is a flow chart showing this process.

Check session status

Set session validation to false

Destroy session and deny further access

Figure 3.12: Flowchart of Logout Process

**3.10 Database Design**

Databases are comprised of tables, tables hold different categories of information. These tables make it easy to query the database. The proposed system database is comprised of three tables namely: user login table, IP registration table and report table.

**3.10.1 User Login Table**

This table contains information regarding user’s login credentials to log into the application. It has fields like serial number, name, username, password, phone number, email. Table 3.5 shows an example.

Table 3.5: User Login Table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Username** | **Password** | **Name** | **Email** | **User level** | **Mobile** | **Status** |
| 1 | Admin | Admin\*\* | Victor | [vic@yahoo.com](mailto:vic@yahoo.com) | Super Admin | 07035323371 | 1 |
| 2 | Cracker | Stego | Nsikak | [nsikak@yahoo.com](mailto:nsikak@yahoo.com) | Supervisor | 08067082189 | 1 |

**3.10.2 IP Registration Table**

This table contains information regarding detailed information on the packet header of a packet detected to be malicious. It has fields like serial number, activity, date and time stamp, source IP, destination IP, source port, destination port, type of protocol (TOP), time to live (TTL). The table 3.6 shows an example.

Table 3.6: IP Registration Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Activity** | **Activity Time** | **Source IP** | **Dest IP** | **Source port** | **Dest Port** | **TTL** | **TTL** |
| 1 | Port scan | 23:36: 11 | 192.168.17.1 | 10.10.0.1 | 25 | 80 | 125 | UDP |
| 2 | Ping sweep | 20:18:12 | 108.101.10.77 | 192.168.25.2 | 88 | 1024 | 75 | TCP |

**3.10.3 Report Table**

This table holds information concerning the result of traceback. It has fields like source network IP, DNS name, location, trace start time, trace stop time, number of hops etc. the table 3.7 shows an example.

Table 3.7: Report Table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Source IP** | **Dest IP** | **Trace time (Secs)** | **DNS** | **Hops** | **Country Location** | **Date processed** |
| 1 | 192.168.0.100 | 10.10.1.12 | 120 | Iphase | 15 | Abuja | 10-07-2012 |
| 2 | 192.168.16.1 | 172.168.17.2 | 117 | FUT | 25 | Minna | 12-07-2012 |
| 3 | 105.108.1.28 | 192.168.31.9 | 90 | VIC | 10 | London | 14-07-2012 |

**CHAPTER FOUR**

**4.0 SYSTEM IMPLEMENTATION, TESTING AND INTEGRATION**

**4.1 Choice of Programming Language**

The designed application is a web-based application which means that it is server-based and can only be implemented on servers (basically on Microsoft servers, from 2003 versions upwards)properly configured and also of high specification in terms of memory and processing. The server is dedicated specifically for the application which in our architecture we called an application server, the application can only be launched using web browsers like Internet Explorer, Mozilla Firefox etc. All unit components of the designed system will be tested for functionality, integrated into a unit system and performance will be evaluated based of specific requirements.

In the design of this system, we considered using pre-hypertext processing (PHP), MySQL (My structured query language), JavaScript, hypertext mark-up language (HTML) and cascading styling sheet (CSS). The choice was based on their individual advantage and because of how best they suit the application. HTML provides feature that makes it easy to design application interfaces that is compatible with most browsers and for the fact that other scripting languages like PHP, JavaScript, Perl, python etc. can be embedded in it to provide some core functionalities and dynamism. CSS was chosen because it provides features that aid in formatting and styling the body of application interfaces, giving it an appealing look and beauty in design.

PHP was chosen because it provides the application with the codes, methods and functions that add the core functionalities needed to drive the application. More so, for the fact that it is a server scripting language, it makes the application easy to load and fast in processing data. It also provides support for object-oriented programming which makes programming very flexible, scalable and maintainable. PHP has functions that can easily and conveniently be used to connect to and pass query to MySQL database.

MySQL is a query language used to design database, it connects easily with PHP and other scripting languages. A MySQL tool known as MySQL Yog is used to design the database, this is because it simplifies database design and reduces the complexity and the number of error encountered in designing databases from the MySQL command console.

JavaScript is a client-side scripting language, it was derived from Java but though they have very different syntax. It was chosen in the development of this application. It provides a means for adding dynamism and animation to the application. It supports OOP and for the fact that it is a client scripting language, it makes communication with the application interfaces very interactive and lively.

**4.2 Systems Drivers/Main Menu Implementation**

The main menu is designed to allow users to easily navigate through the application and perform specific operations in precedence as provided in the user manual. The main menu consists of links that enable the users to perform specific tasks like register IP, trace IP, view report, add user and a logout option. The figure 4.1 shows the implementation of the main menu of the designed system

****

Figure 4.1: Main Menu Implementation

**4.3 Implementation of the Subsystem Drivers/Menus**

The designed system has five basic subsystems namely; the login subsystem, IP registration subsystem, IP trace subsystem, view report subsystem and add user subsystem.

**4.3.1 Login Subsystem Implementation**

Once the web-based application is launched on a web browser, the first interface encountered is the login page. This page provides forms for users to enter a valid username and password. On clicking the login button, access will be granted to authorised users. When an incorrect username and/or password are entered, an error message will be thrown up, prompting for the issuance of a valid username and/or password. If the username and password provided are correct and valid, the user will be redirected to the main menu. The figure 4.2 shows the implementation of the login subsystem.

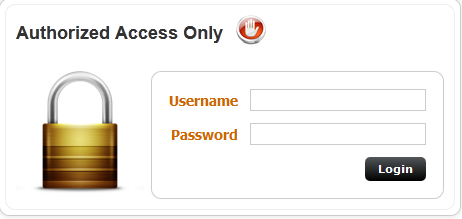
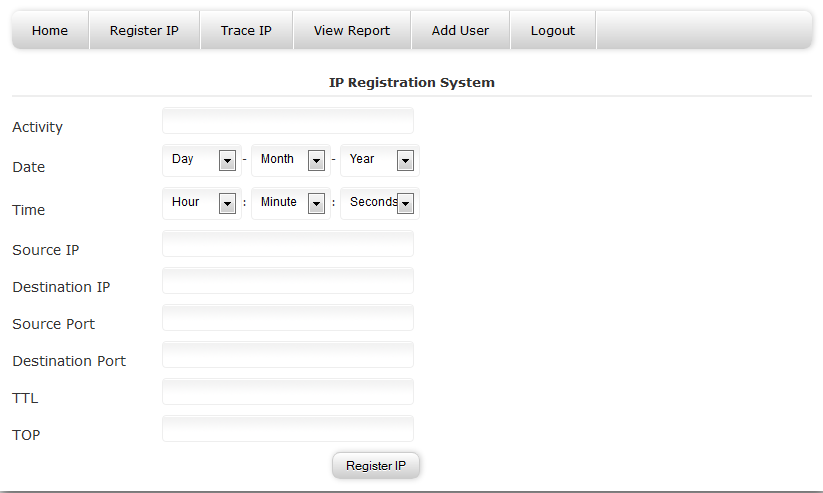


Figure 4.2: Login Subsystem Implementation

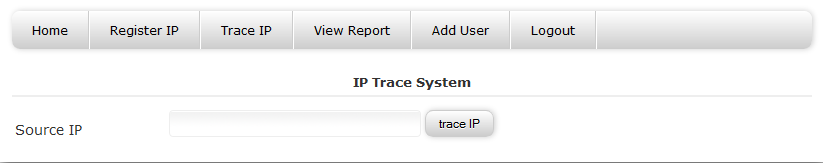
**4.3.2 IP Registration Subsystem Implementation**

This subsystem is designed to store malicious packet header information. This information include: activity, date, time, source IP, destination IP, source port, destination port, time to live (ttl) and type of protocol (top). This information will be collected via the forms provided on the IP registration page. Upon clicking the registered button, this information will automatically be conveyed and stored in the database. The figure 4.3 shows the IP registration subsystem implementation.

 Figure 4.3: IP Registration Subsystem Implementation

**4.3.3 IP Trace Subsystem Implementation**

This subsystem was designed specifically to trace IP addresses. A form is provided where a valid IP address can be entered, when an invalid IP address is entered, the subsystem automatically rejects it and throws up an error message. The trace button, once clicked, runs a back trace on the IP address provided. After trace is complete, the final network address derived as a result of the trace will be mapped to a domain name server (DNS) and a country location. After a full trace, the result will be formatted and displayed on the screen and then pushed into the database for future reference. The figure 4.4 shows the implementation of the IP trace subsystem.

Figure 4.4: IP Trace Subsystem Implementation

**4.3.4 View Report Subsystem Implementation**

This subsystem was designed to make retrieving of results of trace stored in the database easy. On clicking this link, the application connects to the database and queries it to show all the content of trace result stored in the report table. This table has fields like source IP, destination IP, trace time (in seconds), DNS name, number of hops, country location and date and time created. The figure 4.5 shows the implementation of the view report subsystem implementation.

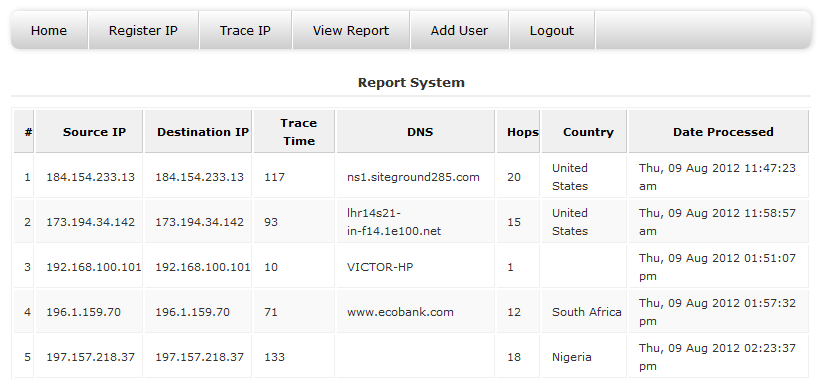


Figure 4.5: View Report Subsystem Implementation

**4.3.5 Add User Subsystem Implementation**

This subsystem was designed to allow users to be created. These users will be legitimate users of the application. Creating users is a privilege available to super administrators and administrators, other users are assigned lower privileges on the system. Privileges are designed and assigned with numbers for easy identification in the database. Super administrators are assigned with a number “1”, administrators are assigned with number “2”, managers are assigned with “3” and supervisors are assigned with “4”. In the database, these assigned numbers are used to define the user level (Ulevel) field, other fields include: first name, last name, username, password, level, email and phone number. The add user button automatically stores the provided information in the user login table in the database. The figure 4.6 shows the implementation of the add user subsystem implementation.

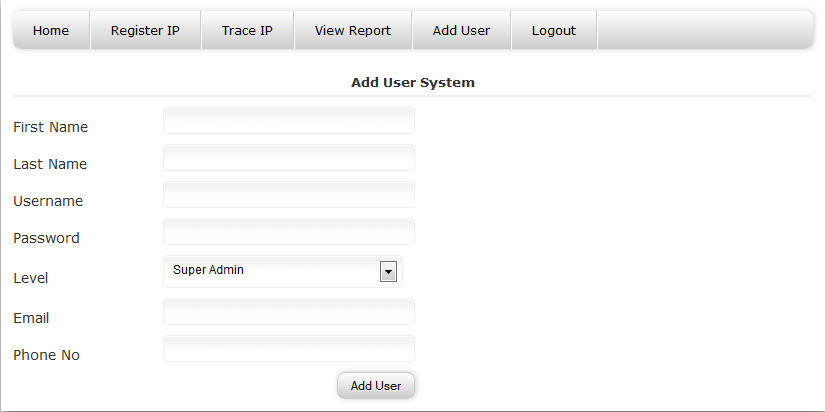


Figure 4.6: Add User Subsystem Implementation

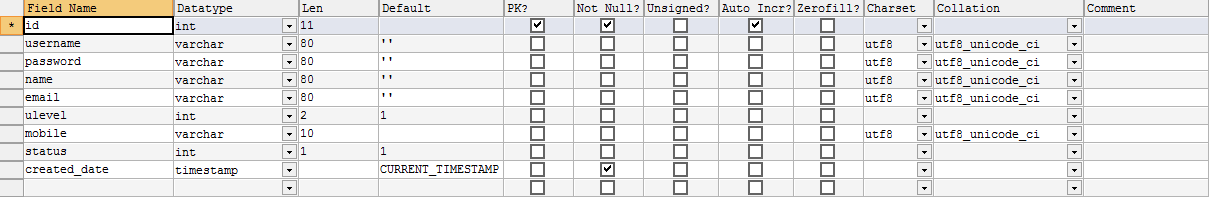
**4.4 Database Implementation**

The database was designed using MySQL. The various table design and their corresponding results will be illustrated with figures and tables. These database tables include: login table, IP registration table and view report table.

**4.4.1 Login Table Implementation**

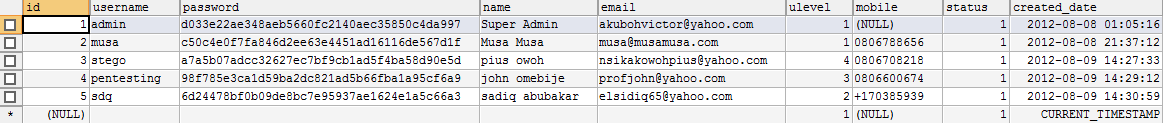
The tables fields are clearly defined, also each field data type and data length are also defined. The table 4.1(a) shows the login table design. The login table data for the login table design in the table 4.1(a) is shown in table 4.1(b). Its fields include: username, password, name, email, user level, mobile, status and date created. In the password field, the password is hashed using Secure Hash Algorithm (SHA 1). The status field is set to “1” just to validate that a user’s detail (especially the username and password) exists in the user login table in the database.

Table 4.1(a): Login Table Design



The table data for table 4.1(a) is shown in table 4.1(b)

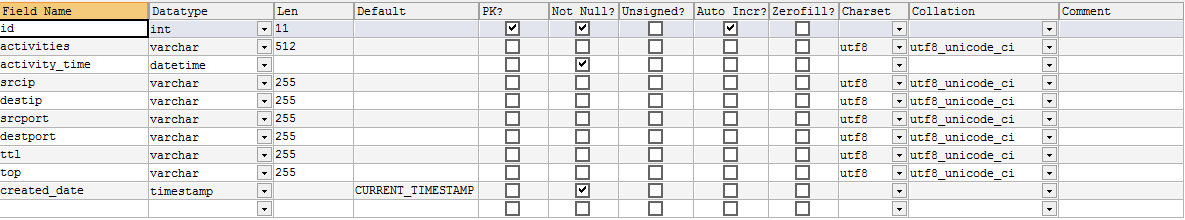
Table 4.1(b): Login Table Data



**4.4.2 IP Registration Table Implementation**

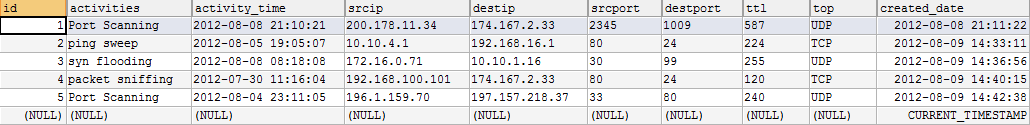
MySQL Yog was used to design the IP registration table, its fields, data type and data lengths are also defined. The fields include activities, activity time, source IP, destination IP, source port, destination port, time to live (ttl), type of protocol (top) and created date. A primary key was assigned and named “Id”. The table 4.2(a) shows the IP registration design. The IP registration table data is shown in table 4.2(b). When IP information is registered at the IP registration subsystem, the data entered is automatically stored in the IP registration table as shown in table 4.2(b)

Table 4.2(a): IP Registration Table Design



The table data for table 4.2(a) is shown in table 4.2(b)

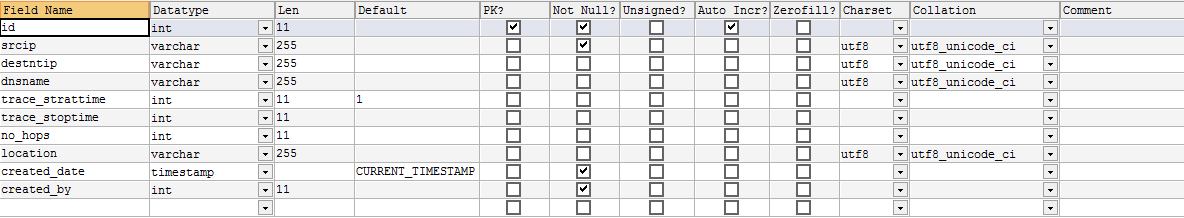
Table 4.2: IP Registration Table Data



**4.4.3 View Report Table Implementation**

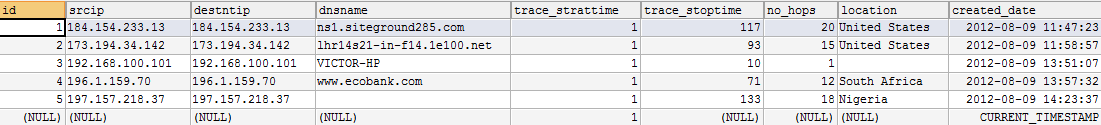
This is the actual design of the report table. This table is designed to keep the result of trace. It has fields like source IP, destination IP, DNS name, trace start time (by default 1 in seconds), trace stop time (in seconds), number of hops, country location and created date, which is the and time the trace was carried out. Each field’s data type and data length is clearly defined. Its primary key is also assigned in numbers. The table 4.3(a) shows the view report design. The table data for the view report table design in table 4.3(a) is shown in table 4.3(b). The result of each trace is stored in this table and can be viewed by authorised persons on the main menu. The table 4.3(a) shows the view report table design.

Table 4.3(a): View Report Table Design



The table data for table 4.3(a) is shown in table 4.3(b)

Table 4.3(b): View Report Table Data.



**4.5 System Testing and Integration**

System testing as the name implies, is simply checking the system components in order to see that they perform task that they designed to do without error. After this component testing phase is successful, the components will then be put together to function in unison as a system. This function is termed system integration. This system can be tested for interior quality; this is an engineering perspective that checks the system for efficiency, test data documentation and structure. Testing can also be for exterior quality, this category of testing is concerned with the system’s correctness, reliability, usability and integrity and lastly, testing can be carried out for futuristic quality, this category of testing is concerned with the system’s flexibility, reusability and maintainability. The designed system has the following components login, IP registration, IP trace, add user, view report and log unit. Each component has been successfully implemented and also their integration as a system was a success.

**4.6 The Test Plan**

System testing is popularly approximated using white-box and black-box testing is carried out to check the internal working of the system, this requires the tester to have programming knowledge. Black-box testing examines the system functionality without any knowledge of the internal working of the system. The tester only needs to be aware of what the system is supposed to do and not how it does it.

In this study, we consider building our test plan in black-box testing; this is to reduce testing complexity and strictly examining what exactly the system is supposed to do. Test will be clearly defined and result will be categorized as “pass” or “fail”. The “pass” and “fail” categories of result are generated from a percentage estimation of test data. It derivation is gathered from the percentage expression using equation (4,1).

Number of successful operation x 100 (4,1)

Total number of operations 1

A maximum error between 0- 20% is acceptable, meaning that the result will be categorized as “pass” its error is between 0- 20% and success percentage is from 80% and above. Result can be categorized as “fail” if its error percentage exceeds 20, that is, success percentage is less than 80%. In this study, we considered testing for the effectiveness of each of the system components. We carried out small scale test of the five (5) operations for each component from which we can calculate our percentage of success or failure depending on the number of successful or failed operation in each of the component out of the five (5) test operations.

**4.7 Test Data**

In this study, test data considered are the basic operation that each system component is expected to perform. These data are; login attempts, IP registration attempts, IP trace attempts, view report attempts, add user attempts and log out attempts.

1. **Login attempts**: Is an attempt that aims at entering login passwords values to

examine if it will be a success or a failure. Its percentage estimation is calculated using equation (4,2).

Number of successful login attempts x 100 (4,2)

Total number of login carried put 1

1. **IP registration attempts**: Like the log in attempts, it aims at entering IP header information to examine if it is successfully registered or stored in the database or not. Its percentage estimation is calculated using equation (4,3)

Number of successful IP registration x 100 (4,3)

Total number of IP registration carried out 1

1. **IP trace attempts**: Examines the system to see if it can successfully trace an IP as claimed. Its percentage estimation is calculated using equation (4,4).

Number of successful IP trace x 100 (4,4)

Total number of IP trace carried out 1

1. **View report attempts**: Examines the system to see if it successfully retrieves the formatted result of IP trace as claimed. Its percentage estimation is calculated using equation (4,5).

Number of successful view report x 100 (4,5)

Total number of view report carried out 1

1. **Add user attempts**: Examines the system to see if it can successfully add a new user to the system as claimed. Its percentage estimation is calculated using equation (4,6).

Number of successful “add user” x 100 (4,6)

Total number of “add user” carried out 1

1. **Logout attempt**: Examines the system to see if it can successfully terminate session as claimed. Its percentage estimation is calculated using equation (4,7).

Number of successful “log out” x 100 (4,7)

Total number of “log out” carried out” 1

The data generated and used for this sample testing for the IP registration attempts, IP trace attempts and add user attempts are shown in table 4.2(b), figure 4.5 and table 4.1(b) respectively. Login attempts’ testing was carried out using the username and password created for users in table 4.1(b). The view report testing and logout require no data for their testing and thus their testing was carried out based on observation.

**4.8 Actual Test Result versus Expected Test Result**

Based on the test data obtained, the table 4.4 shows a comparison between actual test result versus expected test result.

Table 4.4: Actual Test Result versus Expected Test Result

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test data** | **Total No. of attempts** | **No. of**  **successful attempts** | **No. of**  **failed attempts** | **Percentage estimation**  **%** | **Actual test result** | **Expected test result** |
| **Log in**  **Attempts** | 5 | 5 | 0 | 100 | Pass | Pass |
| **IP registration attempts** | 5 | 5 | 0 | 100 | Pass | Pass |
| **IP trace**  **Attempts** | 5 | 4 | 1 | 80 | Pass | Pass |
| **View report attempts** | 5 | 5 | 0 | 100 | Pass | Pass |
| **Add user**  **attempts** | 5 | 4 | 1 | 80 | Pass | Pass |
| **Logout**  **Attempts** | 5 | 5 | 0 | 100 | Pass | Pass |

On the average, the percentage estimated for the test data is:

100 + 100 + 80 + 100 + 80 + 100 = 93.33%

6

The percentage estimate of the test which is approximately 93% indicates that the designed system is quite excellent in terms of performance and functionality. This implies that it possess an error of approximately 7% which is by far very tolerable and provides a window for future modification and enhancement. The result of the percentage estimate of the test is an interpretation that the designed system is efficient, effective, reliable, usable, and maintainable and secure to a great extent and can be recommended for deployment on networks.

**4.9 User Manual**

The guide to using this application is very easy. The following steps will explain in simple terms how this application should be used and strictly in the stated precedence, except for exceptions.

**Step One**: Launch your web browser (say Mozilla Firefox or Internet Explorer).

**Step Two**: Type in the correct user resource location (URL) for the web-based application.

**Step Three**: Enter a valid and correct username and password and then click “login”.

**Step Four**: From the main menu select IP registration and click once on it.

**Step Five**: Fill the form provided on the IP registration subsystem and “register”.

**Step Six**: From the menu bar, click once on IP trace.

**Step Seven**: From the IP trace subsystem, enter the source IP in the form provided and then click “trace” button.

**Step Eight**: Upon display of successful trace feedback message, click “ok”.

**Step Nine**: From the menu bar, click once on view report (to view the results of trace).

**Step Ten**: From the menu bar, click “add users” button.

**Step Eleven**: Fill the form provided on the add user subsystem and then click “add users” button (this operation is for administrators and super administrators only).

**Step Twelve**: Click logout to terminate session (at will anyway).

**CHAPTER FIVE**

**5.0 SUMMARY, RECOMMENDATION AND CONCLUSION**

**5.1 Summary**

At the end of this study, the proposed method of using IDSes behind firewalls and the use of a dedicated application server which houses the designed web-based application that incorporate an algorithm based on ICMP traceback messages, was implemented and tested. Its performance was evaluated and based on that, we summarise and conclude this study by reviewing our achievements, contributions, suggestions, recommendations and conclusions.

Network attacks are a major setback in organizations network and it affects its operations. These activities of network attacks are mainly carried out by malicious attackers and they do these with ease without being caught. We revealed their activities and how they carry them out in phases. On the other hand, network attack detection and traceback is a challenge we also addressed. We discussed IDS, its types and detection schemes and how it can improve the effectiveness of attack detection when placed behind router and firewall in the network architecture. Furthermore, we introduced the major approaches of tracing network attacks and based on their limitations, we proposed a web-based application that incorporates two attack traceback approaches, that is, logging and ICMP traceback message approaches. We also clearly defined the objective, scope, significance and limitation of this study.

We reviewed the concept of network security and also that of Internet Protocol (IP). We analysed the major protocols that is responsible for data transport, basically, TCP and UDP, explaining with figure their header properties. We also reviewed various types of network attacks, illustrating with diagrams how these attacks are carried out. Firewalls, IDSes and traceback approaches was vividly reviewed and illustrated with figures where necessary.

Based on our review, we made a careful analysis of the existing system and spotting out their various limitations based on some criteria. After this analysis, we proposed a method of using IDSes behind firewalls and routers for effective network attack detection and a web-based application. We designed a model for this application and illustrated the kind of network architecture with which it can be properly implemented. We justified the proposed design of the system based on its improvement over previous approaches. Also, a proper design and analysis of all the sub-systems interfaces was done.

We implemented all the sub systems of the designed system, testing all unit components of it based on their success rate, we integrated the system and conducted a performance evaluation considering some functional and non-functional requirements.

Based on our achievements and contribution to knowledge, we summarized, recommended and concluded that our design and methods should be adopted by organizations, although we clearly stated our suggestions for further improvements and modification.

**5.2 Review of Achievements**

In this study, we achieved our objectives which are to prevent storing packet header information on routers which requires large storage and affects router processing and secondly, to reduce the level of ISP involvement in tracing back attacks, which could lead to privacy and legal issues. Another achievement is the improvement of network attack detection brought about by using IDSes behind routers and firewalls to compliment their flaws. In terms of traceback, we achieved to some extent by improving the effectiveness of traceback by introducing a web-based application that allows storing of IP header information on the application server, providing a means for administrators to view malicious IP profile in the database and also to view the result of traceback for future referencing and profiling.

**5.3 Area of Application**

This system can be implemented and very applicable to any networking environment that uses servers, these networking environments are not limited to business enterprises, organisations, government agencies and institutions. Apart from tracing back IPs considered to be malicious, this system can be use to trace just IPs regardless of whether they are malicious or not. Websites can also be traced to their source server, since websites are translated into IP addresses on the internet.

**5.4 Major Contribution to Knowledge**

There are many contributions to knowledge in this study. To start with, network security needs and its components are now clearly understood. Malicious hacker activities have been exposed. A lot about firewalls have been understood, despite its popularity in usage, its flaws were also exposed. We also contributed to knowledge how IDSes use their detection schemes to detect attacks and how IDSes, firewalls and routers can be placed strategically on the network to improve the effectiveness of network attack detection. Types of network attacks and how these attacks can be carried out have been revealed. In tracing back these attacks, we contributed to knowledge the limitations of existing approaches and we introduced a web-based application that can improve the effectiveness of traceback to some extent that previous approaches could not provide, that is, reduction of router storage requirements and involvement of ISP.

**5.5 Suggestions for Further Work**

In as much as the designed system has some advantages, it cannot be 100% effective, it is limited to IP spoofing and steganography, although steganography is still very new, future modification to enable tracing spoofed IP will be a great improvement. Another suggestion is to automatically acquire IP header information from the IDSes rather than manually inputting them. This will reduce the time requirement in keying into the database IP header information.

**5.6 Recommendation**

After careful analysis, design, implementation, testing and evaluation of the system, it can be recommended that business enterprise, organization, government agencies and institutions can implement the use of this web-based IP trace system (W-BIS) as proposed in the design and network architecture. This is because it provides an architecture that can effectively detect network attacks, an application server dedicated to the application, where packet header information can be stored, results of trace can be viewed for future referencing. All these feature provided improves router processing, improves network speed and reduces the level of ISP involvement in attack traceback.

**5.7 Conclusion**

It can be concluded in this study that network security is more expensive compared with mere implementing a network, but considering the havoc network attacks cause to organizations, the cost of securing network pay off. Network attack traceback is a very important aspect of network forensic analysis, this is because it provides information is admissible in a court of law and can lead to the arrest of perpetrators. With the challenges of ISP involvement and router processing involved in attack traceback, the designed system. When implemented by organizations will lead to a more effective and efficient way of tracing back network attacks, thereby improving the speed and performance of network forensic analysis.