Network Traffic Threat Detection and Reporting System Validation through UML

Amit Kumar Bhardwaj[1] Maninder Singh[2]
1.L.M. Thapar School of Management, Thapar University, Patiala, Punjab-147004, India
2.Computer Science Engineering Department, Thapar University, Patiala, Punjab-147004, India
akbhardwaj@thapar.edu;msingh@thapar.edu

Abstract
In today's digital world, computer network security experts struggle to manage security issues effectively. Reporting the network data in graphical form helps the expert to take decision in more effective and efficient way. Visualizing the network traffic seamlessly is a big challenge but an integrated network traffic visualization approach can resolve such issues effectively. The work presented here focuses on structural, behavioral and architectural modeling of an Integrated Network Traffic Visualization System (INTVS) and validating it through unified modeling language. The adopted modeling can accommodate the analysis and designing of INTVS effectively, which is demonstrated in this study.

Keywords: Network traffic visualization, Network Security, INTVS framework, INTVS modeling.

1. Introduction

Data Visualization gives a holistic view of network through graphical representation of various types. As it is well said "a picture depicts a thousand words in itself" by Shneiderman B. (1996), this approach will lead to quick interpretations and actions. These actions provides the dataset of network traffic in a graphical form, which further helps to network administrators in decision making and to understand the network security phenomenon for quick response. Withall et al. (2007). classified the known network data visualization schemes in three categories viz. geographical, abstract and plot based visualization and most of the these visualization schemes are using three main modules: capturing , parsing and visualization. Most of these schemes are using third party tools like, Netflow and Wireshark are used as capturing module. Other side the INTVS framework proposed by Bhardwaj et al. (2014), a novel approach of network traffic visualization, which works in real time environment to capture the network traffic, tokenize, parse and visualize it seamlessly, as INTVS is equipped with its own module (capturing module, tokenizing module, parsing module and visualizing module). Even INTVS is able to work in both wireless and wired networks. In INTVS, the output facilitates its user to understand the data patterns while displaying and reporting the malicious traffic patterns through different visualization schemes. Before implementing INTVS, it is designed and validated using UML. As Booch et al. (1999), Jacobson et al. (1999) and Rumbaugh et al. (2005) discussed in details, the usage of UML to model and validate the real world problem/system using forward engineering.

Further the organization schemes of this research study as follows. *Section 2* discuss the literature of main VizSec (Visualization based network security solutions) and utility of UML to model and validate Information Technology (IT) solution. *Section 3* outlines the limitation of the present visualization tools. *Section 4* deals with INTVS framework and *Section 5* outlines the validation of INTVS framework through UML. *Section 6* demonstrates the implementation of INTVS and its potential uses. Finally, conclusion is presented in *Section 7*.

2. Related Work

This section briefly describes the network traffic visualization tools and role of UML in validating the different IT solutions.

2.1 Literature review of popular network traffic visualization tools:

Flodar is visualization tool developed by Swing (1998), which gives a high-level view of the network and servers within a network. Estrin et al. (2000) developed Nam, which is capable of showing the packet-level animations and these animations are usually from 'ns' simulation package and generate trace files. NVisionIP tool developed by Lakkaraju et al. (2004), is used for visualizing entire Class B network on a single screen. Yin et al. (2004) complete the work of Lakkaraju et al. (2004) effectively utilizing the capabilities of NVisionIP and VisFlowConnect-IP together and ending with SIFT (Security Incident Fusion Tool). VISUAL (Visual Information Security for Administrator Live) was developed by Ball et al. (2004). VISUAL is capable in visualizing the connectivity between private network hosts and 10,000 external hosts in 3D grid square display. Estan et al. (2005) developed Wisconsin Netpy, helpful to evaluate the network traffic at run time.
Abdullah et al. (2005) developed IDS Rainstorm, can display 163,830 IP addresses on a single visualization, using pixels and colours to represent alarms for a large network. Conti et al. (2007) developed a tool named as

Rumint to meet the overloaded information of security. Rumint can analyze 30,000 packets (maximum) at a time. AfterGlow 1.x (developed in PERL) developed by Marty (2008). AfterGlow uses pcap (packet capture) files and log files to send e-mail as input data sources. Afterglow 2.x, uses treemap visualization scheme to display the logs of various services effectively.

Reil et al. (2006) developed InetVis to analyze the suspicious activity in internet traffic under a class-C category. Flamingo is a client/Server based network security visualization tool developed by Oberheide et al. (2006). Flamingo can represent the live data, while using OpenGL dataset and uses data from Netflow (version 5).

FloVis developed by Taylor et al. (2009), supports heterogeneous environments of operating systems (Windows/ Linux/ Unix). VIAssist developed by Goodall et al. (2009), a visualization tool developed on Microsoft.Net framework. to have an overview of the network with the help of an intuitive dashboard in interactive manner.

Makanju et al. (2009) developed the LogView to visualize the log of application layer services like IMAP, POP3, SSH and HTTP. Jiawan et al. (2009) developed NetViewer, capable in detecting the DDoS attacks, network scans, and port scans.

2.2 UML Modeling for IT Solution Validation based literature

Jong (2002) revealed designing of an embedded real-time systems for a telecommunication application while showing the applicability of the flow to control and data -dominated types of systems, through UML and SDL. Ming et al. (2002) introduced Integrated network solution to manage multiple technologies and domain networks, efficiently and cost effectively. Kukkala et al. (2004) developed and implemented a medium access control protocol named as TUTMAC while validating it with the help of UML. Ray et al. (2005) used UML to reveal the attack model and its defense mechanism.

Wenhui et al. (2007) used the UML to validate the VPN service management system. Dwyer et al. (2008) demonstrated the network layout and details thorough visualization, based on UML class diagram and biological networks. Devamalar et al. (2008) used UML to validate the web centric intelligent health care diagnosis system WEBDIACIN. Sekaran et al. (2009) validate the network protocols for their implementation over software and / or hardware in an efficient manner through UML Lipsinky et al. (2009) used UML, for space reliability modeling formalism of distributed systems and service complaints.

3. Limitation of Discussed Visualization Tools

The current visualization schemes are lacking in integrated network traffic visualization tool, and the still there is scope of validating such IT solution though UML. So far, as per our knowledge no work has been done in the past in the direction of INTVS validation through UML modeling. Further to validate the static, architectural and behavioral aspects of such systems, present study contains INTVS validation through UML modeling followed by its experimental results.

4. Framework of INTVS

There are six functional modules works for INTVS as shown in Figure 1. The first module capture the network traffic, second module carrying out tokenization of the data according to network traffic properties, third module perform the parsing, forth module used the parsed file to generate the visualization, fifth module is dealing with real time data analysis with the help of data mining schemes and last module helps in forensic analysis of the network data with the help of data mining. The INTVS is using multi-threads to handle these modules effectively.

5. Designing and Validating INTVS through UML Diagrams

This section discuss the designing and validation of INTVS through three fundamental UML modeling techniques propounded by Booch et al. (1999), Jacobson et al(1999), and Rumbaugh et al. (2005).

5.1 The use-case diagram INTVS is shown in Figure 2 and showing the dynamic aspect of internal and external influence of INTVS to identify the functionality & actors. The external actors are N/w admin and network users; and internal actors are modules and databases of INTVS. The functionalities required by N/w admin are defining network policy (which includes the network traffic rules for ingress and egress traffic, bandwidth allocation rules for all users as per network policy), selecting media, N/w monitoring (to understand the state of network whether normal or under attack). Further, N/w admin would like to control the network resources, while taking some corrective decision to protect the network resources from mischievous activities to ensure the effective utilization of the network. Another external actor is network user, who want to understand whether his or her node/ machine is under attack or not, for this he or she can select the media, view the capturing of raw data, see various network information in a visual form and can take decision with help of functionality offered by the INTVS. For internal actor, INTVS database accepts the inputs and returns the results to the system. The INTVS use-case diagram specifies the above said events of INTVS and their flows in a specific manner and produces the high-level view of the INTVS for its users.

5.2 Activity Diagram

In Figure 3, the *Admin, User, INTVS and DATABASE* system are identified as swim lane. The activities performed by the *Admin* are updating N/w policy w.r.t time, creating N/w policy database, selecting media (wireless or wired), monitoring visualization and decision making. The *USER* group (swim lane) performs various activities such as selecting media, monitoring visualization and decision making. Swim lane *INTVS* performs various activities such as: getting inputs from admin and users, capturing live data, tokenizing the N/w data, parsing the N/w data, validating N/w traffic for malicious activity using knowledge base system, visualizing N/w traffic data, exporting data for forensic analysis, Forensic analysis through visualization (here INTVS helps in Forensic analysis of network traffic for future usage). The fourth swim lane *DATABASE* has the following activities: N/w policy database, maintaining tokenized data and updating it. INTVS activity diagram can also be used to construct the executable system by using forward and reverse engineering techniques.

5.3 Class Diagrams

There are four classes shown in Figure 4 and termed as *User_Type*, *INTVS*, *Media* and *N/w Policy*. INTVS class diagram also describes the functionalities performed by various sub-systems. *INTVS* class diagram leads to the construction of INTVS network security application using object oriented language. Even as *INTVS* class diagrams gives the base for component and deployment. Further, *User_Type* is generalized as user and *N/w admin* sub-class. *N/w admin* and *User* sub-class possess the same attributes, but *N/w admin* is having more operations as comparison to *User (*shown in the Figure 4). *INTVS* class defined the various operations: taking input( ), Capturing( ) - both operation Tokenizing( ), Parsing( ), Data mining( ), Visualizing( ). *N/w policy* class posses three attributes, Rule_id, Rule_name and Rule_type ; and having two operations Policy formulation( ) and Policy update( ). Class *Decision* posses the decision_id, and decision_name attributes; and having three operations: allow( ), drop( ), Update( ). Class *Media* posses the Type_of_Media attribute. Class *N/w details* is subclass of INTVS having following attributes srno_of_pkt, SIP, DIP, src_port, Dst_port, pkt_length, pkt_window_size, pkt_session, pkt_timestamp, pkt_protocol. Subclass *Visualizing* posses the Graph_id and Graph_name attributes; and performing Display View( ) and Display Customized View( ) operations.

5.4 Sequence Diagram

In Figure 5, Sequence diagram is used to visualize the sequence of messages generating in INTVS to perform specific functionalities. For example, object *:N/w Admin* defines the network rules while sending the message "Define_Rules" to object *:INTVS*, then *:INTVS* object send these details about the network resources and their utilization rules to *:N/w Policy Database* object and so on a functionality of defining network rules and network policy is accomplished. To update the rules as per the need of time, another functionality is initiated while sending a message "Update_Rules" from *:N/w Admin* object to *:INTVS* object. Then same is communicated by *:INTVS* object to *:N/w Policy Database* object through "Get_updates" message. Another functionality is, selecting a media initiated by *:N/w Admin* object and send a message "Select_Media" to *:INTVS* object. Then *:INTVS* object take this call and start capturing network data live from selected media, while sending a self message "Capturing_N/w_live_data". On similar way the functionality of tokenizing and parsing are accomplished by *:INTVS* object. The functionality of detecting any malicious activity and generating alert is done by *:INTVS* object after receiving the call from *:N/w Policy Database* object. Then *:INTVS* object sequence the next functionality of sending parsed data to database for real time and forensic analysis. The functionality of generating various customized view of network traffic accomplished when *:N/w admin* take the decisions of sending a call to *:INTVS* object "Visualize_graph", then object *:INTVS* gives the choice for any specific view. A view is selected by *:N/w_Admin* object, then a selected view is displayed by *:INTVS* object. Based upon view (which is summarizing an event) a decision is taken by *:N/w_Admin* object, so the functionality of how a decision suppose to be taken by the network administrator/user is accomplished. These interactions among the components of INTVS are very important for implementation and execution perspective.

5.5 Collaboration Diagram

The second interaction diagram is collaboration diagram, which shows object organization of INTVS (Figure 6). In collaboration diagram, methods are called one after another as per the sequence numbered.

5.6 Statechart Diagram

Statechart diagram of INTVS defines different states of an object during its lifetime. The statechart diagram is useful to model reactive systems, in which the INTVS is responding to external or internal events.. Identifying the important objects is the primary job in statechart diagram and in case of INTVS network traffic/ data is identified as an object, which moves from one state to another like captured raw data to tokenized data, then tokenized data to parsed data, then parsed data to visual report form as shown in Figure 7. Meanwhile some events are triggered which causes the changes in of states of network data. This includes INTVS selecting media, capturing, tokenizing, parsing, alerting, data mining, visualizing and decision making. When selecting any media, an event is triggered next event is capturing the data (which is in signal form) state to convert in raw data states. As soon as raw data generated, event tokenized takes place immediately, and converts raw data into tokenized data and changed the state of object from raw data to tokenized data. As soon as INTVS find the tokenized data, another event is triggered named as parsing, which causes the changes in tokenized data in parsed data state. Generating alert based on altering event, analyzing data according to network policy, and if not valid, generating an alert, another state of data in signal form. Then parsed data is mined (based on data mining event) and parsed data converted in mined parsed data. As soon as parsed data generated in INTVS, the visualization event is triggered and parsed data converted into visual report state for decision making.

5.7 Component diagram of INTVS

Component diagrams are used to model physical aspects of INTVS as shown in Figure 8. The INTVS Component diagrams are used to visualize the organization and relationships among components of its system.

5.8 Deployment diagram

The static deployment view of INTVS is shown in Figure 9. The purpose of INTVS component diagrams is to describe the components whereas deployment diagrams shows how these are deployed in hardware (as shown in Figure 9). The INTVS deployment diagram is showing its hardware topology, describing the hardware components used for software component (system software or application software). It also, describes the runtime processing nodes of INTVS.

6. INTVS Implementation for Forensic Analysis

Implementation of INTVS deals with its coding, testing and installation. For the INTVS operational working on Windows platform, it requires Winpcap and JRE; where as in the case of Linux, Ubuntu and Solaris, user would need to have libpcap and JRE. INTVS uses libraries as: JCommon-1.0.17, JfreeChart-1.0.14 and jgrapht-jdk1.6, which helps to develop the visualization engine to produce various visualizations w.r.t. network traffic data. The parsed file in XML format is used for real time traffic analysis, to avoid the extra middle layer as required in RDBMS for analysis. The parsed file in CSV and .xls format are used for forensic analysis and for network resource monitoring. The partial outcome of INTVS was discussed in our previous work [30].

During implementation we categorized the attacks in three major category w.r.t. network, transport and application layer.

At network layer each packet is analyzed for its source Internet protocol (SIP), destination Internet protocol (DIP) address according to network policy for ingress and egress traffic. There are certain nodes and VLAN (virtual local area network) such as server farm/nodes (email server, Web kiosk server, database server) which are particularly accessible by particular nodes purposely, rest are prohibited. If any such node, which is not privileged to access server farm node from ingress and egress, then an attack is identified and classified under N/w layer attack (sniffing and spoofing attacks). Generally, the campus area network (CAN) and VLAN experience the denial-of-service (DoS) and distributed denial-of-service (D-DoS) attacks, for this we have used the IP-spoofed tool to experiment.

At transport layer, transmission control protocol (TCP) and user datagram protocol (UDP) are used to establish node to node connection for data transmission. At server side, there are certain ports open according to N/w policy for its various users, if anybody try to violate these rules or try to deceive the node and its ports, then *Transport layer attack* and *Transport layer attack (Deceptive)* are observed respectively. These attacks are identified based on SIP, DIP, source port (S_port) and destination port (D_port) combinations.

Generally at application layer, many attacks are experienced w.r.t. email service (SMTP Mail Flooding, spamming), HTTP server (HTTP-based attacks spanning multiple packets, HTTP header spoofing attacks), FTP service (FTP bounce attack, passive FTP attacks, client and server bounce attacks, FTP port injection attacks) etc. These are identified based on various features of packet such SIP, DIP, S_port, D_Port, Pkt_lenght, Session and timestamp.

We experiment the INTVS in an isolated environment for forensic analysis to take corrective measures through a data mining scheme. We launched different attack knowingly, to capture such data to validate our model for the network layer attack, transport layer attack as well as application layer attack. Further, the forensic analysis of network traffic is done by INTVS as reported in a dashboard form (Figure 10). It explains the current scenario of the network and helps to formulate the future network policy decisions. In Figure 10, the dashboard presents Total Time, Total Number of Machine, Total number of VLANs, Total data transmitted, Total Intranet load, Total Internet load, VLANs based traffic, Particular VLAN based load over machines, Particular machine load, Reporting load, protocol-wise w.r.t. Campus area network, VLAN, Machines, and different types of attacks.

Figure 10 (a) describes different types of attacks that are detected by INTVS at network, transport and application layer. INTVS have been experimented on a machine (2GB RAM, Intel Core™ 2 Duo processors) for two minutes and 13629 data packets have been captured. There are 62 nodes and 39 VLANs in operations are observed. In this time span, total data transmitted is 67960 Kb under 92 data sessions from 18 network layer attacks and 11 application layers attacks detected. Figure 10 (b) describes the data transmission and it is found that only internet based traffic is observed. Out of 39 VLANs the *VLAN 172.31.19.0* is highly overloaded. In CAN there are 10 different network services observed and it is found that HTTP is maximum used. Figure 10 (c) describes the load of all the detected machines of a VLAN (172.31.19.1), and it is found that maximum resource consuming machine is 172.31.19.184, and also found that machine (172.31.19.184) is using maximum HTTP service.

7. Conclusion

The work presented here emphasizes the role of visualization based network security solution to handle the network security issues more precisely and accurately. Here, INTVS system demonstrates a novel approach of network traffic visualization, which is able to work in both wireless and wired networks. The INTVS also works in real time environment to capture the network traffic, tokenization, parsing and visualization. The output facilitates in understanding the data patterns, while displaying and reporting the malicious traffic patterns through different visualization schemes. Different UML modeling techniques are used to validate the INTVS framework for structural, behavioral and architecture modeling. The INTVS as a network security solution is well documented and presented through different experiments. The INTVS implementation is tested for different scenarios in monitoring and controlling the network security issues. The study validates that the proposed INTVS is effective in reporting the utilization of network resources and helps to formulate the new network policy. The present work can be extended for visualizing the cloud computing services. It can also be used for mobile based applications to address its security issues.

References

Swing, E. (1998). Flodar: Flow Visualization of Network Traffic. Computer Graphics and Applications, IEEE , 6-8.

Lakkaraju, K., Yurcik, W., Lee, A. J., Bearavolu, R., Li, Y., & Yin, X. (2004). NVisionIP: Netflow visualizations of system state for security situational awareness. Workshop on Visualization and Data Mining for Computer Security (pp. 65-72). ACM.

Ball, R., Fink, G.A., North, C.R. (2004). Home-centric visualization of network traffic for security administration. Workshop on Visualization and Data Mining for Computer Security (pp. 55-64). ACM.

Yin, X., Yurcik, W., Treaster, M., Li, Y., Lakkaraju, K. (2004). VisFlowConnect: NetFlow Visualizations of Link Relationships for Security Situational Awareness. Workshop on Visualization and Data Mining for Computer Security (pp. 26-34). ACM.

Estan, C., Magin, G., (2005). Interactive Traffic Analysis and Visualization with Wisconsin Netpy. Conference on Large Installation System Administration Conference (pp. 177-184). USENIX Association.

Abdullah, K., Lee, C.P., Conti, G., Copeland, J.A., Stasko, J. (2005). IDS RainStorm: Visualizing IDS Alarms. Workshop on Visualization for Computer Security, IEEE (pp. 1-10).

Reil, J.Jean-Pierre van RielP.V., Irwin, B. (2006). InetVis, a Visual Tool for Network Telescope Traffic Analysis. AFRIGRAPH, (1-59593-288-7/06/0001). ACM.

Oberheide, J., Goff, M., Karir, M. (2006). Flamingo: Visualizing internet traffic. Symposium on Network Operations and Management, IEEE/IFIP (pp. 150-161).

Taylor, T., Paterson, D., Glanfield, J., Gates, C., Brooks, S., McHugh, J. (2009). FloVis: Flow Visualization System. Conference on Cybersecurity Applications and Technology, Homeland Security (pp. 186-198).

Goodall, J.R., Sowul, M. (2009). VIAssist: Visual Analytics for Cyber Defense. Conference on Technologies for Homeland Security, HST '09, IEEE, (pp. 143-150).

Jiawan, Z., Liang, L., Liangfu, L., Ning, Z. (2008). A Novel Visualization Approach for Efficient Network Scans Detection. International Conf. on Security Technology, SECTECH, IEEE (pp. 23-26).

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy of information visualizations. Symposium on Visual Languages, IEEE (pp. 336-343).

Estrin, D., Handley, M., Heidermann, J., McCanne, S., Xu, Y., Yu, H. (2000). Network visualization with Nam, the VINT network administrator. IEEE Computer

Conti, G., (2007). Security Data Visualization: Graphical Techniques for Network Analysis, San Francisco: No Starch Press.

Marty, R., (2008). Applied Security Visualization, Boston: Addison Wesley; 1 edition.

Makanju, A., Brooks, S., Zincir-Heywood, A. N., & Milios, E. E. (2008). LogView: Visualizing Event Log Clusters. Sixth Annual Conference on Privacy, Security and Trust (pp. 99-108). IEEE Computer Society

Jiawan, Z., Peng, Y., Liangfu, L., & Lei, C. (2009). NetViewer: A Visualization Tool for Network Security Events. International Conference on Networks Security, Wireless Communications and Trusted Computing (pp. 434-437). IEEE Computer Society.

Bhardwaj A.K., and Singh M.,(2014). Data mining-based integrated network traffic visualization framework for threat detection. Neural Computing and Applications Journal, Springer, DOI : 10.1007/s00521-014-1701-2.

Booch, G., Rumbaugh, J., and Jacobson, I. (1999). Unified Modeling Language User Guide. Addison Wesley.

Jacobson, I., Booch, G., and Rumbaugh, J. (1999). The Unified Software Development Process. Pearson Education.

Devamalar, P. M., Bai, V. T., Murali, N., and Srivatsa, S. K. (2008). Visualization and Construction of Real Time Web Centric Intelligent Health Care Diagnostic System Using UML. CCECE/CCGEI (pp. 501-506). Niagara Falls. Canada: IEEE.

Dwyer, T., Marriott, K., Schreiber, F., Stuckey, P. J., Woodward, M., and Wybrow, M. (2008). Exploration of Networks Using Overview+Detail with Constraint-based Cooperative Layout. IEEE Transactions On Visualization And Computer Graphics. 14, pp. 1293-1300. IEEE Computer Society.

Hu, S. X., and Shan, T. C. (2010). Designing Resource Oriented Architecture in UML - A Case Study on Smart Grid Home Area Network (HAN). IEEE 6th World Congress on Services (pp. 154-155). IEEE.

Jong, G. d. (2002). A UML Based Design Methodology for Real- Time and Embedded Systems. Design, Automation and Test in Europe Conference and Exhibition. IEEE Computer Society.

Ming, H., Hong, W., and Luoming, M. (2003). Solution and Architecture for Integrated Network Management. Proceedings of ICCT2003, (pp. 1650-1654).

Kukkala, P., Helminen, V., Hannikainen, M., and Hamalainen, T. D. (2004). UML 2.0 Implementation of An Embedded WLAN Protocol. IEEE , 1158-1162.

Ray, H. T., Vemuri, R., and Kantubhukta, H. R. (2005). Toward an Automated Attack Model for Red Teams. IEEE SECURITY and PRIVACY , 18-25.

Rumbaugh, J., Jacobson, I., and Booch., G. (2005). Unified Modeling Language Reference Manual (Vol. 2nd Edition). Pearson Education.

Wenhui, S., Feng, L., Gang, D., and Jinyu, Z. (2007). Formal Analysis of the VPN Service Management System. Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies (pp. 493-497). IEEE Computer Society.

Withall, M., Phillips, I., Parish, D. (2007). Network visualization: a review. IET Commun., Vol. 1, No. 3 (pp. 365-372).

Sekaran, K. C., and Gnanamurthy, R. K. (2009). Executable Specification and Prototyping of Network Protocols Using UML and Java.

Lipsinky, Z. (2009). UML-based reliability modeling of network services, a UDP Echo service case study. Fourth International Conference on Dependability of Computer Systems (pp. 50-57). IEEE Computer Society.

Authors:

Amit Kumar Bhardwaj is assistant professor in the LM Thapar School of Management, at Thapar University Patiala, Punjab, India, His research interests include network security, data and information security, information system analysis and design, e-business and digital marketing. Amit Kumar received his Master of Engineering in software engineering from Thapar University Patiala. Contact him at akbhardwaj@thapar.edu.

Dr. Maninder Singh (Head, CITM) is associate professor in the Computer Science and Engineering Department at Thapar University. His research interests include network security and grid computing, and he is a torchbearer for the open source community. Singh received a Ph.D in network security from Thapar University. Contact him at msingh@thapar.edu.

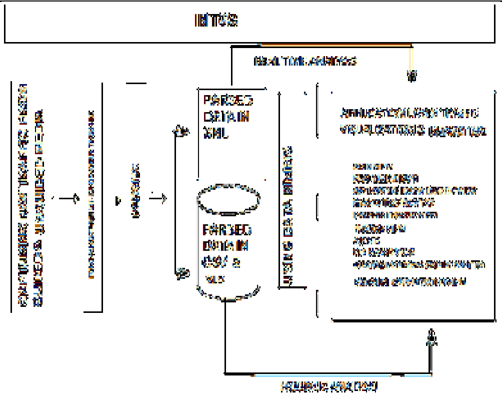


Figure 1: Framework of INTVS (Source: Proposed by authors)

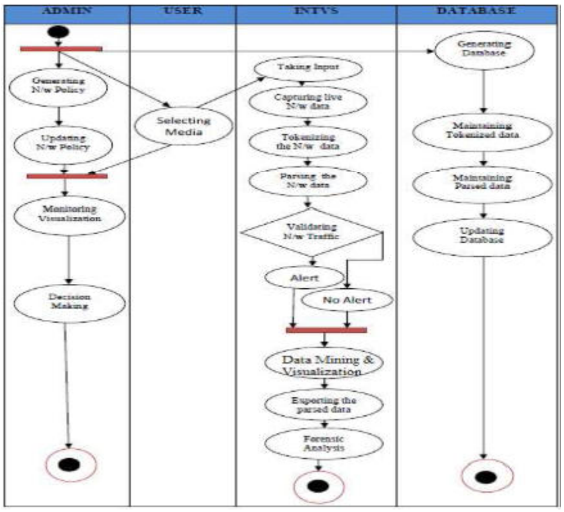


Figure 3: Activity diagram of INTVS

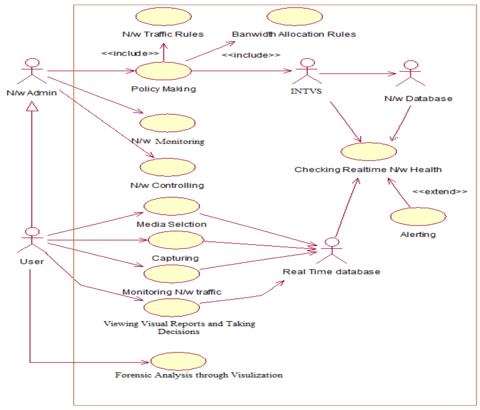


Figure 2: Use case diagram of INTVS

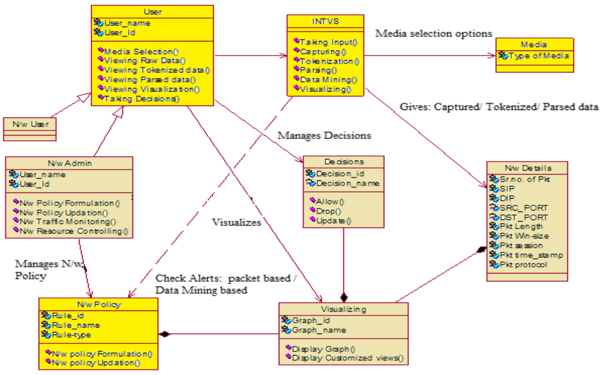


Figure 4: Class diagram of INTVS

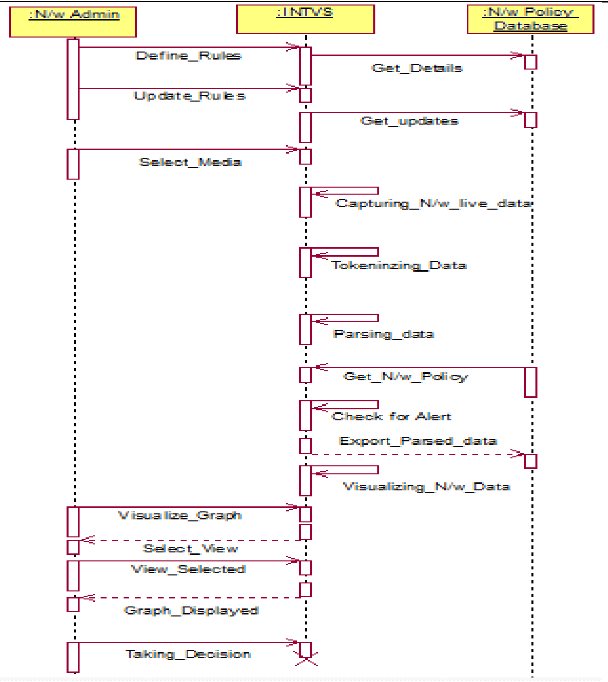


Figure 5: Sequence diagram of INTVS

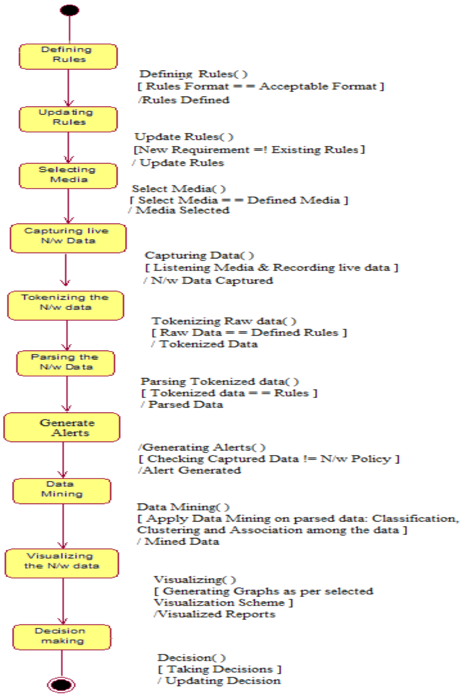


Figure 7: State-chart diagram of INTVS.

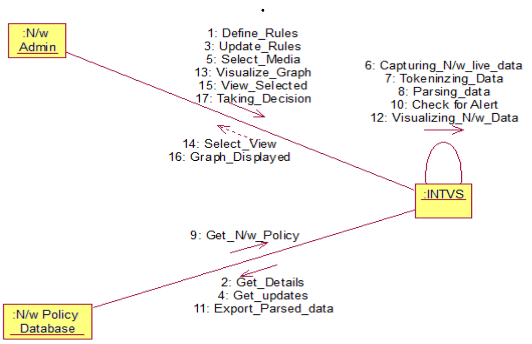


Figure 6: Collaboration diagram of INTVS

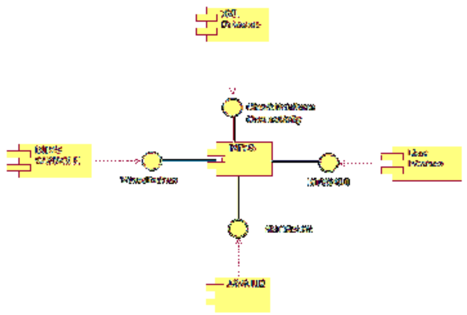


Figure 8: Component Diagram of INTVS.

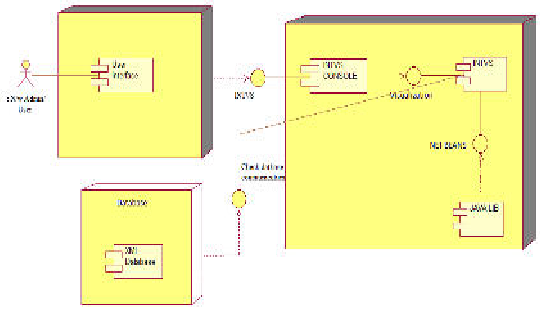
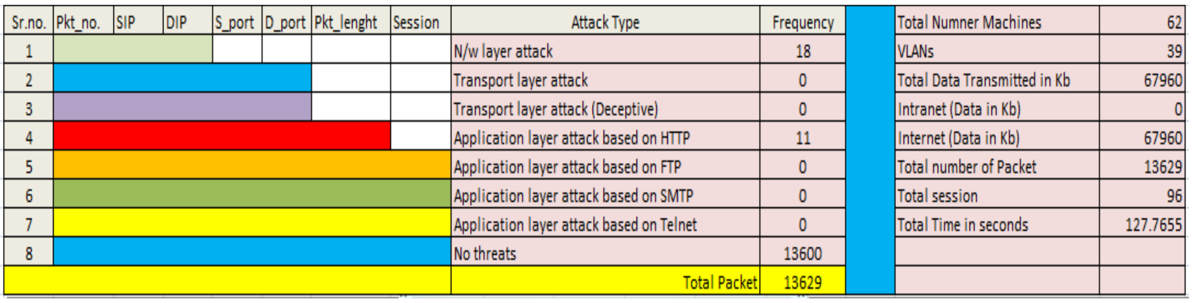
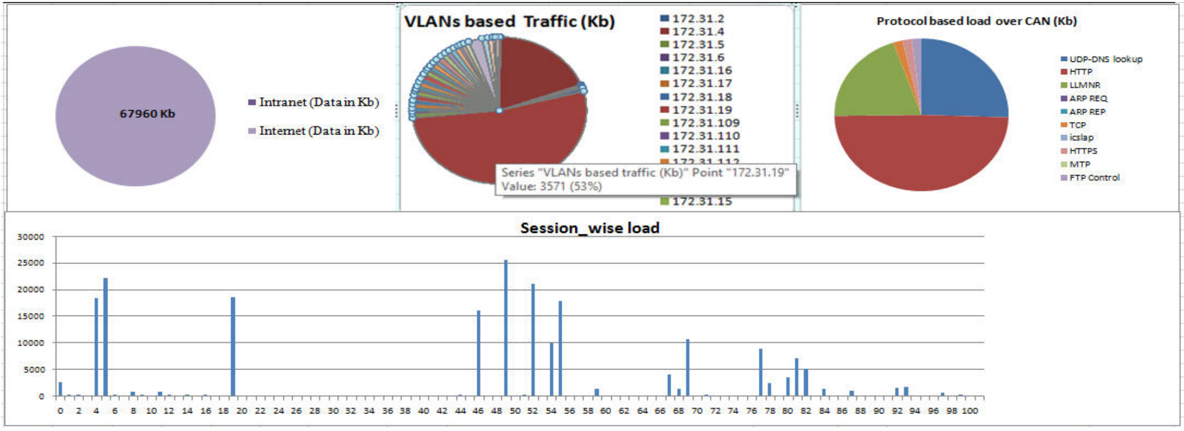


Figure 9: INTVS - Deployment Diagram

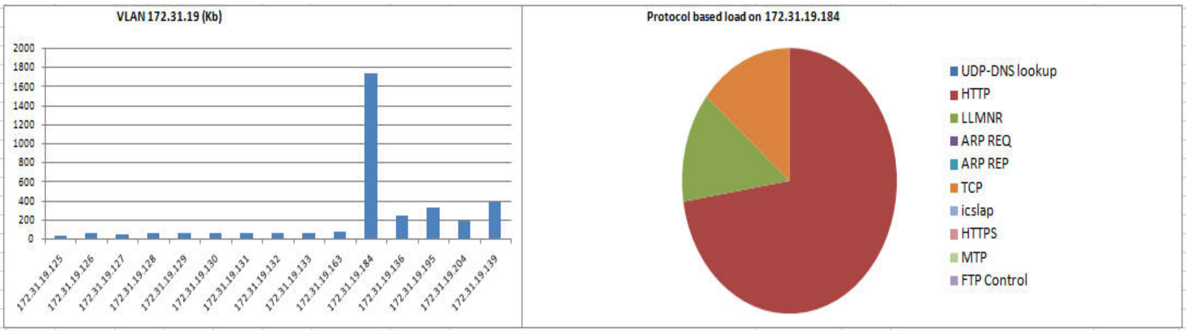


Sr.no.	Pkt_no.	SIP	DIP	S_port	D_port	Pkt_lenght	Session	Attack Type	Frequency		Total Numner Machines	62
1	🟩	🟩	🟩					N/w layer attack	18		VLANs	39
2	🟦	🟦	🟦	🟦	🟦			Transport layer attack	0		Total Data Transmitted in Kb	67960
3	🟪	🟪	🟪	🟪	🟪			Transport layer attack (Deceptive)	0		Intranet (Data in Kb)	0
4	🟥	🟥	🟥	🟥	🟥	🟥		Application layer attack based on HTTP	11		Internet (Data in Kb)	67960
5	🟧	🟧	🟧	🟧	🟧	🟧	🟧	Application layer attack based on FTP	0		Total number of Packet	13629
6	🟩	🟩	🟩	🟩	🟩	🟩	🟩	Application layer attack based on SMTP	0		Total session	96
7	🟨	🟨	🟨	🟨	🟨	🟨	🟨	Application layer attack based on Telnet	0		Total Time in seconds	127.7655
8	🟦	🟦	🟦	🟦	🟦	🟦	🟦	No threats	13600			
								Total Packet	13629			

(a)



(b)



(c)

Figure 10: Forensic Analysis (Source: Result of INTVS)