Neural Comput & Applic (2015) 26:117–130
DOI 10.1007/s00521-014-1701-2

ORIGINAL ARTICLE

Data mining-based integrated network traffic visualization framework for threat detection

Amit Kumar Bhardwaj · Maninder Singh

Received: 5 March 2013 / Accepted: 10 August 2014 / Published online: 31 August 2014

Abstract In this speedy and voluminous digital world, the threat detection and reporting are a challenging job for rapid action. The present study deals with a strong and viable solution to overcome different threats, network security using data mining approach and techniques through visual graphical representation. Current research study explained and proposed a novel approach named as 'integrated network traffic visualization system'. Nevertheless, current framework is working and based on data mining, further help out to demonstrates two new visualization schemes called as: Firstly Grid and secondly Platter. Per framework results, the Grid view is capable of displaying network traffic in different classified grids, based on application layer protocols. Additionally, Platter view visualizes campus area wireless network traffic on a single screen mechanized automatically adjusted with network size. These active schemes are significantly effective to identify and monitor the compromised machines and cuts down reaction time.

Keywords Data mining · Grid view · Integrated network traffic visualization system · Platter view

A. K. Bhardwaj (✉)
L.M. Thapar School of Management, Thapar University, Patiala 147004, Punjab, India
e-mail: amitsep30@gmail.com

M. Singh
Computer Science Engineering Department, Thapar University, Patiala 147004, Punjab, India

1 Introduction

Information visualization has become an integral part of network security tools, as it is well said that, "a picture depicts thousands words [1]". Human brain responds rapidly to a picture than text because the eye and the visual cortex of the brain form a massively parallel processor that provides the highest bandwidth channel into human cognitive centers [1]. Numerous commercial, private, public and social web applications generate voluminous network traffic over Internet. The Internet users are observing bitter experience as computer machines; websites and network resources are in trap under different network attacks. In order to avoid these network attacks, many security researchers [2–5] have outlined and highlighted the usage of data mining techniques as it helps in classification, clustering, association and summarization. Data mining-based network measures are gaining importance in security field, and result is that many tools and techniques have started to incorporate data mining approaches at various levels of security like in firewalls [6], intrusion detection systems (IDSs) [7] and intrusion prevention systems. The approaches mainly highlight text-based responses and have their own limitations [6, 7]. The information visualization is one of the most intuitive and fast security approaches. This helps network administrator to get the traffic patterns in graphical form for instant analysis and response.

The scheme of research paper as follows: Sect. 2 is about the contribution of researchers in the field of network traffic visualization, usage of data mining in network security visualization for network threat detection and situation awareness. Gap analysis attributed in next Sect. 3. In the direction of integrated network traffic visualization system (INTVS), the workflow and its methodology are

discussed in Sect. 4. Designing and implementation of INTVS discussed in Sect. 5.

2 Related work

In this section, we are discussing the work of many key researchers in three different domains of information security. (1) Security through data visualization, (2) security through data mining techniques, and (3) security through data visualization based on data mining. Swing [8] developed a tool, named as Flodar, able to display high-level view of the network and servers available within a network. In this scheme, data servers shown by rectangular standing bars in different circles, eventually servers are steadily pushed toward the center of the Platter. However, information about open systems interconnection (OSI) model layer, media and data sources are not mentioned in this work. Estrin et al. [9] developed a tool named as 'Nam'. This tool runs in offline mode. 'Nam' is capable of showing the packet-level animations from network simulator (ns) simulation package, which generates traces. These trace files contain required information for static network layout and for dynamic events. This tool is equipped with VCR like buttons, which are helpful to view a particular portion of the data.

Yin et al. [10] developed 'VisFlowConnect', a network security tool, which is a supplement part of SIFT. The main purpose of 'VisFlowConnect-IP' is to represent the flow between internal networks visually. It is also facilitating filtering and drill down. Yin et al. used CANINE to synchronize the different NetFlow formats (the Cisco NetFlow and Argus NetFlow). VisFlowConnect used parallel coordinate plots to display the connectivity between different machines based on IP address. There are three parallel vertical axes used, firstly central axis represents the IP address of source machines, and the other two axes used to display destination and subnet machines IP address.

Ball et al. [3] developed a tool named Visual Information Security for Administrator Live (VISUAL). The VISUAL is capable in visualizing the connectivity between centric network host and 10,000 external hosts in 3D grid square display. VISUAL is also equipped with multiple port view, interactive filtering based on protocol (TCP, UDP and ICMP) and time. VISUAL is using tcpdump and ethereal for data source and generating 3D grid square display. However, VISUAL is limited to network layer visualization only. No real-time data visualization is possible.

Fink et al. [11] proposed a network visualization tool named as 'Portall', a prototype of a Network Eye framework. This prototype is an extended version of previous prototype of Network Eye framework of VISUAL. Kim and Reddy [12] developed the 'NetViewer', there are five main component of 'NetViewer', the packet parser, the signal-computing engine, the detection engine, visualization engine and alert engine. 'NetViewer' used libpcap, Cisco's NetFlow and NLANR as input source. 'NetViewer' is capable to produce the real-time picture of the network traffic with the help of images to detect and identify malicious traffic patterns based on multiple dimensions of network traffics such as IP addresses, port numbers, and the number of flows.

Estan et al. [13] developed Wisconsin 'Netpy' an extended version of Netpy in Politehnica University of Bucharest with collaborative efforts of Computer Science of the University of Wisconsin-Madison USA, to evaluate the network traffic in run time. 'Netpy' is having four main components: graphical user interface (GUI), database, console and analysis engine. It uses NetFlow data source and stores data in database for further analysis. The GUI is developed using wxPython UI tool kit. 'Netpy' is an interactive tool for analysis of source addresses, destination addresses, filters and time series plots that separate the traffic into various categories. This tool supports the run time interactive drill-down facility for network layer protocol, port and time-based filters.

Abdullah et al. [14] proposed IDS Rainstorm to overcome the problem of text-based evaluation of log files of IDS at Georgia Institute of Technology. The IDS Rainstorm can display 163,830 IP address on a single visualization while using pixels and colors to represent alarms. This tool is able to represent network layer-based connectivity and their alarms according to network policy. This tool facilitates to see the details of a particular IP by filtering and zooming. The source of input of this tool is alert logs generated by Snort. IDS Rainstorm works offline and displays scattered plot and parallel coordinate plot visualization.

Conti et al. [15] also developed tool named as 'Rumint' to meet the overloaded information of security. 'Rumint' can analyze 30,000 packets (maximum) at a given time. This tool is capable of visualizing binary file and display port vulnerability assessment reports. *Rumint* is also helpful in dissecting port scans, Nessus vulnerability assessments and Metasploit attacks. This tool is able to display the logs of firewall and IDS. VCR like animation facility makes this tool more effective to interact with historical data in effective manner through scatter visualization.

Marty's [16] developed 'AfterGlow' using PERL and input sources are pcap files and log file of sendmail. 'AfterGlow2.X' developed in java uses Treemap visualization scheme to display email Logs, firewall Logs, firewall rule sets, web logs, IDS logs, IPS logs and operating system logs. It produces comma-separated values (CSVs) file after parsing the tcpdump. This CSV file is used to produce dot file as an output, and this dot file further is

used as input for Graphviz. Graphviz is an another tool used to produce graphs of the logs using spring models. Graphviz also facilitates its user to view data in real time of a specific domain/area linked with problem. Moreover, Marty expanded this work and proposed the Data Visualization and Analysis Linux (DAVIX) framework. DAVIX [17] is equipped with 25 tools, which are freely available for data visualization.

Reil and Irwin [18] developed 'InetVis' to analyze the suspicious activity present in Internet traffic for C-class networks. 'Nmap', libpcap and tcpdump are used as input source to capture the network data. InetVis is capable to visualize the ICMP and UDP packets in real time. InetVis facilitate user with adjustable time window, navigation (for deeper exploration) and dynamic filtering. It can replay data while adjusting its rate. However, InetVis is limited to visualize the network layer data only.

Oberheide et al. [19] developed 'Flamingo', a client/server-based tool, which is capable of displaying the live data while using OpenGL dataset. Flamingo uses parallel axis concept to represent individual IP address and their respective load by using bar height. This tool also represents the load per port-tree square layouts at different heights along the cube.

Decker et al. [20] developed NfSen, using various plugins to detect the mischievous data in NetFlow based on PHP (front-end plugin). NfSen works on network layer. NfSen facilitate its users with filtering, aggregation, customized time window, statistic summary to understand the properties of NetFlow data, which focusing on flows of data, packets and bytes, Nfsen is helpful to detect DoS, and DDos attacks.

Godinho et al. [21] developed 'PRISMA' a multiple coordinated views information visualization tool, developed in JAVA. PRISMA includes three visualization techniques: Treemap, Scatterplot and Parallel Coordinates.

'FloVis' developed by Taylor et al. [22] is based on client and server architecture and supports windows, Linux and Unix platforms. It uses SiLK repositories to store the NetFlow and IDS alerts. In FloVis, there are three different kind of visualizations: FlowBundle, Activity and NetBytes viewer. The purpose of Activity viewer is to display a selected part of sequential behavior of a random group. The 'FlowBundle' displays the flow of interactions between hosts or subnets in bundles. The NetBytes viewer is used for detailed behavior analysis of an individual machine based on time parameters.

Allen and McLachlan [23] developed 'NAV' Network Analysis Visualization in JAVA to visualize the high-level network events, to investigate the malicious activities for large network, while operating on network and transport layer visualizations. The input source of this tool is log files with IP address of high-level devices. NAV facilitates to have overview and detailed view of network traffic with IP wall view, Port scan view, service view, open dialog, textual filter, time filter, and color brushing feature for drill down.

Goodall and Sowul [24] developed 'VIAssist' a visualization tool, developed using Microsoft's.NET framework. This tool produces big-picture overview of the network with the help of an intuitive dashboard. The drill-down facility helps in selecting a particular event and displaying it with multiple visualizations with the help of automatic Smart aggregation.

Jiawan et al. [25] developed 'ScanViewer' designed to capture large-scale ports information. ScanViewer can detect hidden scans along with network scans, port scans and distributed port scans. Osborne et al. [26] developed an 'EIC framework' to investigate digital forensic information from a large volume of data by using tool named 'Infovis'. Infovis is equipped with zooming, filtering and visualization facility.

Lu et al. [27] proposed a network scan tool named as 'CCScanViewer'. They conclude that polycurves display is better than parallel coordinated view after experimenting with different datasets.

Vaarandi [7] proposed data mining-based IDS solution for real-time alerts generator against threats. Golnabi et al. [6] used data mining to update rules of firewall. Ahmad et al. [4] used artificial neural network (ANN) based on multiple layered perceptron to detect attacks over the network. Data source for their work was Kddcup99 dataset. They used the backpropagation for training and testing of their system. ANN is having advantage over other data mining techniques for data cleansing, flexibilities of making association, sequence and linkage among the networks. Nonetheless, Ahmad et al. did not provide a visualization-based solution.

Lakkaraju et al. [2] developed 'NVisionIP', a visualizing tool using data mining, for entire Class B network node visualization on a single screen. It facilitates drill down and collects information in more details about a particular host in a network. There are three different levels of visualization galaxy view, small multiple views and machine view. NVisionIP uses Argus NetFlow data and Cisco NetFlow data as its input data source.

Creese et al. [28] proposed a situational awareness-based visualization tool named as 'CyberVis'. This tool is able to visualize the enterprise network attacks and their subsequent potential consequences combining traditional network diagram icons with business process modeling and notation. This tool facilitates drill down and helps in forensic analysis using movie-style playback.

Singh et al. [29] used k-means clustering algorithm to analyze and visualize the flow of data to predict threats based on three attributes of data flow, i.e., protocol, IP and ports. They used third party tool for capturing data and

Table 1 Network traffic visualization tools: a comparative view with INTVS

Tool	Media	Real time	OSI layer	UI	VizSec
Flodar [8]	NM	No	NM	No	Platter display, 3D spinning cube, building display
Nam [9]	Both	No	Up to 7th	Yes	Link graph, monitor boxes, time event graph
VisFlow-Connect [10]	NM	No	3rd and 4th	Yes	Parallel axes
SIFT [10]	NM	Yes	3rd and 4th	Yes	Scatter plot and Parallel coordinated plot, fish eye, glyph
VISUAL [3]	NM	No	3rd and 4th	Yes	3D grid square display
PortAll [11]	Wired	No	3rd and 4th	Yes	Time series graph, Scatterplot, 3D bar graphs
NetViewer [12]	NM	Yes	3rd and 4th	Yes	Simple X–Y graphs and real-time traffic visualization
Wisconsin Netpy [13]	NM	Yes	3rd	Yes	Time series plots and Heatmaps
IDS Rainstrom [14]	NM	No	3rd	Yes	Scatter plot and parallel coordinated plot
Rumint [15]	Both	Yes	3rd	Yes	Binary rainfall, scatter plot, parallel coordinate plot
AfterGlow [16]	Both	Yes	7th	None	Network graph, Treemap
InetVis [18]	NM	Yes	3rd and 4th	Yes	3D scatter plot
Flamingo [19]	NM	Yes	3rd and 4th	Yes	Parallel line, bar graph in quad-tree structure
NfSen [20]	NM	Yes	3rd	Yes	Time series graph
Prisima [21]	NA	NA	NA	Yes	Treemap, scatter plot and parallel coordinates
FloVis [22]	Both	Yes	3rd and 4th	Yes	Activity viewer, FlowBundle, NetBytes viewer
NAV [23]	NM	Yes	3rd and 4th	Yes	Line graph, Netmap
VIAssist [24]	NM	No	3rd	Yes	Histograms, parallel coordinated plot, glyphs
NVisionIP [2]	NM	No	3rd and 4th	Yes	Bar charts for small multiple view, machine view
INTVS	Both	Yes	7th	Yes	Grid view, listmap view, line graph, Platter view, Pie charts

NM not mentioned, *NA* not applicable, *VizSec* visualization scheme, *UI* user interactive

used 'Weka' a data mining tool for analysis. Table 1 given below compares the various network traffic visualization tools with INTVS. The parameters used for the comparison includes tool name, media, real time, OSI layer, User Interactive (UI) and Visualization Scheme (VizSec).

3 Gap analysis

In the above section, discussed tools exhibit limited scope like in Flodar, the researcher did not discussed about the data source, media and OSI layer. Wisconsin Netpy, IDS Rainstrom, Rumint, FloVis, VIAssist are capable to visualize network layer data only. NVisionIP, VISUAL, VisFlowConnect-IP, SIFT, InetVis, Flamingo, and FloVis; all these tools are limited to network and transport layer data visualization. In Nam, there are flaws with respect to traceability of lost packets and real-time visualization, even though it is capable of visualizing application layer data. AfterGlow is capable to perform forensic analysis of application layer data, but is dependent upon a parsing tool, which will parse the log file in CSV format.

Based on review of these tools and framework, research does not report any tool/technique, which is capable to produce interactive visualization of network traffic of large network in seamless manner. These visualization systems lack in one way or other to capture packets, tokenize, parse and then visualize them in an integrated interactive manner up to application layer data in real time. This float a requirement for INTVS, which is required to capture the live packets from guided–unguided media, can tokenize and parse the data for analysis and classification. Framework should be able to display the correct status of all active nodes of a network through visualization in real time for application layer traffic. Proposed work INTVS (*a framework, which will work in both unguided and guided media, captures network traffic, tokenizes, parses and visualizes up-to 7th layer with visualization schemes to detect the threats and reporting in real time*) offer solutions to the mentioned gaps and further explained in Sect. 5.

4 Integrated network traffic visualization system (INTVS) framework

Herby, proposed INTVS framework, is having six main functional modules as shown in Fig. 1. These modules are classified as follows: *First module* is a network traffic capturer (which helps to capture network data). *Second module* is tokenization module, will do the tokenization of the data according to network traffic properties. *Third module* is parsing module, will do parsing of the network traffic to use as an input for visualization engine, which is a *Fourth module*. Similarly, *Fifth module* is dealing with

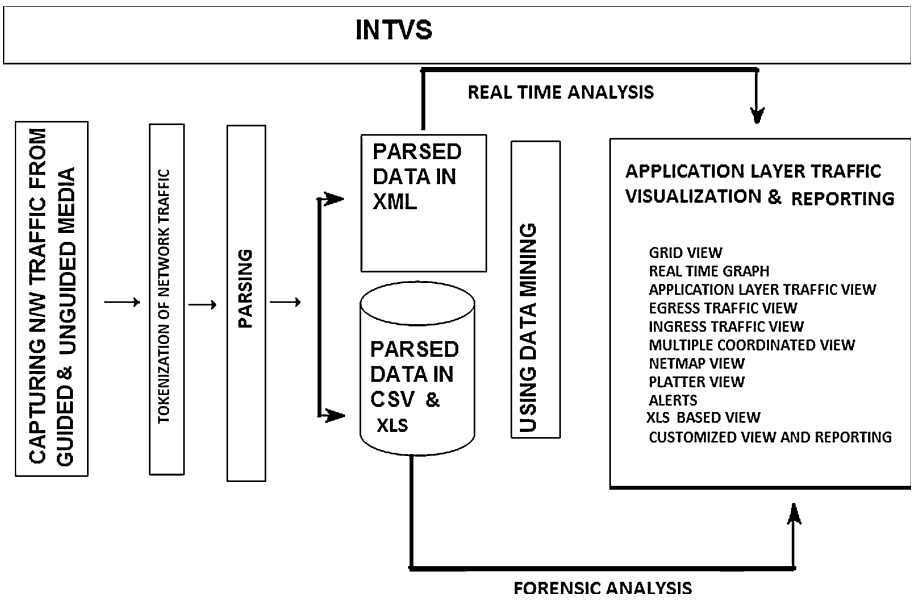


Fig. 1 INTVS framework layout

real-time data analysis with the help of data mining techniques, and *Sixth module* is helpful in forensic analysis of the network data, it also uses data mining techniques. INTVS framework is using multi-threads to handle these modules effectively. For INTVS, memory (RAM) and speed (processor) are prime concerns to handle a big data set, in experimentation framework. These issues related to framework were taken appropriately to address various configurations.

4.1 Methodology

In the study, a methodology followed to support the INTVS framework consists of three prime layers as shown in Fig. 2, (1) input layer, (2) processing layer and (3) output layer. In present research, a hybrid approach of data mining also used to analyze network traffic patterns effectively. To manage the network resources, a supervised data mining technique applied to cater the situation. Network policy considered as an input for this processing layer. On the other side, un-supervised data mining technique applied to learn data pattern through clustering and association methods. Suitable statistical analysis of these data patterns is helpful in detecting and reporting threats.

4.1.1 Input layer consists of following attributes

1. Live packet—Captured from guided and unguided media.
2. Network policy—Details about resource authorization and availability for privileged users and guest users.

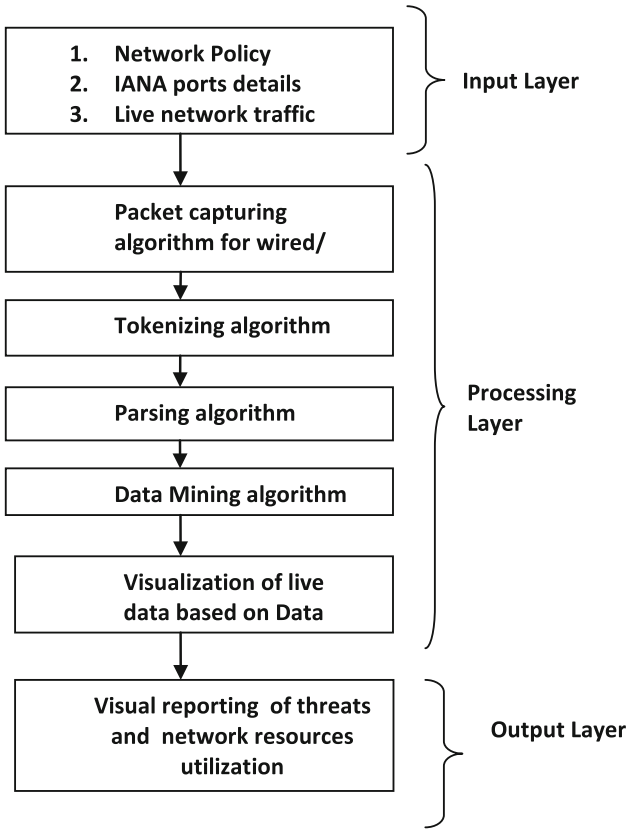


Fig. 2 INTVS workflow

3. Standard port numbers [Internet Assigned Numbers Authority (IANA) numbers]—A list used for classification of network traffic.

4.1.2 *Processing layer is involved with following steps in sequence*

1. Capturing module captures raw data (packets).
2. Data cleansing performed by analyzing and tokenization of attributes.
3. Parsing of cleansed data performed for real-time and forensic analysis.
4. In this step, data piped to real-time analysis stub as well as forwarded to visualization engine.
5. Next, classification algorithm used to classify packets based on NetFlow, Internet, Intranet, TCP, UDP and application layer protocols like Mail protocols, FTP and HTTP. This classified data also forwarded to visualization engine.
6. Clustering of packets is performed based on various parameters like uploads, downloads, total NetFlow per IP, particular VLAN-based traffic segregation, Intranet & Internet traffic flow based on application layer protocols. This helps in performing Machine-Level traffic analysis.
7. Next, association rules applied to establish relationship among key attributes.
8. Finally, threat detection performed and visualized.

4.1.3 *Output layer generate the following reports*

This section deals with visualization engine to display demanded and customized views, to understand, analyze and respond to various conditions during attacks. Following visualization scheme is outcome of Output layer along with mouse-over and drill-down facility.

1. Grid view
2. Platter view
3. Listmap view
4. Real-time analysis graph
5. Alert log file
6. Two-dimensional Network and Machine-Level View

4.2 Salient feature of the INTVS

1. *Media Support*—INTVS is capable of capturing network data from both guided and unguided media, a feature that offered by limited tools such as Nam, Rumint, AfterGlow and FloVis.
2. *Interoperable*—INTVS is compatible to diverse variety of operating systems such as Windows, Linux, Solaris etc. as it is developed using Java.
3. *Real-time and forensic analysis*—In real-time domain, there are very few tools that offer real-time analysis until today. The INTVS is parsing the captured traffic data in XML format using for real-time analysis mode. INTVS is also capable in handling forensic analysis of network traffic while exporting the data to relational database management system (RDBMS) for offline reporting and analysis.
4. *Security analysis of application layer traffic*—As discussed in Sect. 2, only Nam and AfterGlow are capable to analyze application layer data that too with limitations. However, the INTVS can facilitate visualization of application layer data in addition to tracking of the network, machine and type of web service and all related stats like volume of data uploaded and downloaded. Further based on the traffic pattern, this framework can detect the attacks and anomalies.
5. *Visual information*—INTVS framework learns its traffic pattern and generates alerts according to the customized settings. It follows the principles of Shneiderman [30]. In Grid view and Platter visualization, INTVS facilitate its user to customize data filtrations, drill down to reach the source of the problem. In addition, the system can also select a particular node for analysis. Data mined two-dimensional analysis view is very helpful in application layer traffic analysis, which contributed toward novelty in network traffic visualization.
6. *Scalable*—INTVS framework supports real time as well as off line data analysis in addition to supporting multi format data. The system has the capability to capture and visualize traffic from a small network of hundreds of nodes to thousands of nodes.
7. *Intelligence*—It is necessary to understand the traffic patterns and take necessary action for self-defense in the absence of the manual actions as corrective measures. This is an aspect that most of the tools till-date is unable to facilitate. INTVS framework is capable to manage huge visualization data on a single screen, thus administrator to identify and display compromised machines in a network through colors schemes of dot/ring and circle in a Platter visualization mode. Algorithm to detect malicious activity in a network reported in Sect. 5.
8. *Reporting*—INTVS proposes data patterns reporting via Platter visualization and two-dimensional traffic analysis. It can respond many queries such as—how many VLANs are present, how many hosts are live, Downloads and Uploads by single host, port association, protocol wise bandwidth utilization etc. This system also helps network security analysts to view intended information and piping it to visual reports like network bandwidth consumed by particular application layer traffic with reference to total traffic. INTVS is portable, scalable and facilitates its users to export the captured data for future references in CSV and.xls files for forensic analysis.

The operation of this system on Windows requires Winpcap and Java Runtime Environment (JRE) where as in the case of Linux, Ubuntu and Solaris, user would need to have libpcap and JRE. INTVS uses JCommon-1.0.17, JfreeChart-1.0.14 and jgrapht-jdk1.6 libraries, used in developing the visualization engine to produce various visualizations. The parsed file is in XML format used for real-time traffic analysis to avoiding delay due to an extra middle layer required for analysis as far as RDBMS data storage is concerned.

5 INTVS design and its implementation

Design and implementation of various modules of INTVS given and detailed as follows:

5.1 Algorithm for capturing the network traffic

Algorithm -1: Capturing
(Source: Proposed by Authors)

1. BEGIN
2. Devices[] is a 1D array that is used to store all the available n/w devices of the machine
3. Capture is an object use to store the jpcap utility provided by application layer for capturing the packet
4. CapturePacket is a 1D array that is used to store the captured packet
5. Device holds the selected device by user
6. Initialize the Capture object for selected device
7. Start capturing using loopPacket functionality of capture
8. while(true) go to step4 when receivedPacket
9. CapturePacket.add(receivedPacket)
10. END

Moreover, the implemented results of INTVS capturing algorithm outlined in Fig. 3, in support of media selection and real-time capturing network traffic.

The INTVS can capture network data in real-time environment as shown in Fig. 3, which is more or less similar to available Wireshark capturing tool (open source) in the global market.

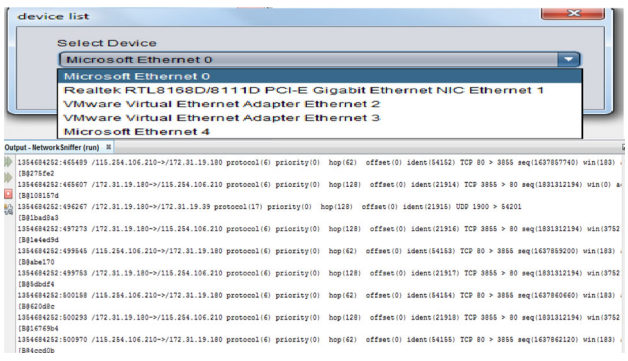


Fig. 3 Selection of media and real-time capturing. (*Source*: INTVS results)

5.2 Designing and implementation of tokenize module

The INTVS can tokenize data and can use it for parsing in real-time environment. Multi-threads programming concepts used to accomplish the required job in parallel. Thus, dedicated multi-threads involved through Tokenizing and parsing, etc. The output of capturing module is used as input to tokenization module.

Algorithm - 2: Tokenization
(Source: Proposed by Authors)

1. BEGIN
2. character[][] is a 2D char array in which character[i] holds the raw captured packet character[i][j] holds a character of raw packet
3. token[][] is a 2D array in which tokenized data is stored after extracting packets from character[][] token [i] represent a captured packet number , token[i][j] represents the indented attributes which are extracted from character[][]
4. repeat through step 23 for each character[i] packet in character[][]
5. if character[i][0]='A' set token[i][5]='ARP' otherwise go to step 6
6. while character[i][j] != ':' increment j
7. set token[i][0] = read from character[i][j+1] till character[i][k] != '/'
8. while character[i][l] != '/' increment l
9. set token[i][2]= read from character[i][l+1] till character[i][m]!=' '
10. while character[i][m] != '(' increment m
11. set token[i][5] = read from character[i][m+1] till character[i][n]!=')'
12. while character[i][n] != 'P' increment n
13. set token[i][3] = read from character[i][n+1] till character[i][o]!='>'
14. while character[i][o] != '>' increment o
15. set token[i][4] = read from character[i][o+1] till character[i][p]!=' '
16. while character[i][p] != 'seq(' increment p
17. set token[i][7] = read from character[i][p+1] till character[i][q]!=') '
18. while character[i][q] != 'win(' increment q
19. set token[i][8] = read from character[i][q+1] till character[i][r]!=')
20. while character[i][q] != ack ' increment r
21. set token[i][8] = read from character[i][q+1] till character[i][s]!=' '
22. while character[i][q] !=length ' increment s
23. set token[i][8] = read from character[i][q+1] till character[i][s]!=' '
24. END

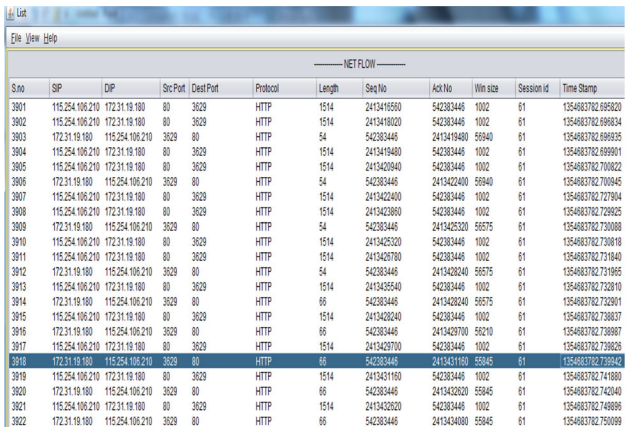


Fig. 4 Tokenized data in real time. (*Source*: INTVS results)

After implementation of tokenization module, the outcomes are shown in Fig. 4. The tokenized data are further parsed in two types of formats, XML for real-time analysis and XLS for forensic analysis.

5.3 Data mining module of INTVS framework

Algorithm 3: Data Mining for Classification of entire traffic i.e. Netflow etc (into OSI application layer protocol) *(Source: Proposed by Authors)*

1. BEGIN
2. token[][] is a 2D array that stores the tokenized packet where token[i] represents a particular token, and, token [i][j] represents information about that token i.e. its SIP (token[i][0]), DIP(token[i][1]), SRCPORT(token[i][2]), DSTPORT(token[i][3]), SEQNO(token[i][4]), ACKNO(token[i][5]), WINSIZE(token[i][6]), SESSIONID(token[i][7]), TIMESTAMP(token[i][8])
3. PORT_DESC[][] is a constant 2D array, as per IANA standards, where PORT_DESC[i] represents a port, and, PORT_DESC[i][j] represents its attributes port no, port description
4. classified_packets[][][] is a 3D array, which stores classified packets, where classified_packets[i], represents a particular protocol (HTTP, FTP, SMTP, OTHERS), classified_packets[i][j] represents a token of type classified_packets[i], and classified_packets[i][j][k] stores the details of packet classified_packets[i][j]
5. Convert PORT_DESC[][] into a hashmap PORT_HASH (Key - Value pair)
6. set desc_temp := PORT_HASH.get(token[i][2])
7. if desc_temp not equals "others" set token[i][9] := desc_temp, else goto Step 8
8. set token[i][9] := PORT_HASH.get(token[i][2])
9. set classified_packets[j][k] := token[i] (According to protocol)
10. END

Here, data mining algorithm for traffic classification in INTVS implemented and these results shown in Fig. 5. The entire network traffic classified as follows: NetFlow, Internet, Intranet, TCP, UDP and HTTP traffic grids along with different color background.

Similarly, INTVS facilitates customization of Grid view component instead of using NetFlow, Intranet, Internet, TCP, UDP and HTTP traffic view; wherein user can select Grid view of NetFlow, Intranet, Internet, Email, FTP and HTTP traffic view as shown in Fig. 6.

In addition, INTVS also provide the extended facility of selecting and filtering a particular traffic view as shown in Fig. 7, where it is showing the view of HTTP/HTTPS traffic. However, user can also view FTP, Email and other application layer traffic in this case.

INTVS also maintains listmap of NetFlow data of the network while continuously updating it. Simple mouse drag option is very useful, when hovered over listmap it shows the traffic stats of a particular machine, like uploading/download data size, TCP data, UDP data and email. Listmap can easily viewed from Fig. 7 left-bottom section where machines of the network are detected and listed in view of HTTP/HTTPS Traffic. Real-time Graph is another way to present the NetFlow of network in a wave form. This helps user to easily understand the data patterns in the network traffic view. In Fig. 7, right bottom graph with colors: yellow, purple, red, blue, green and turquoise line curve are representing HTTP, Email, TCP, UDP, FTP and other traffic, respectively.

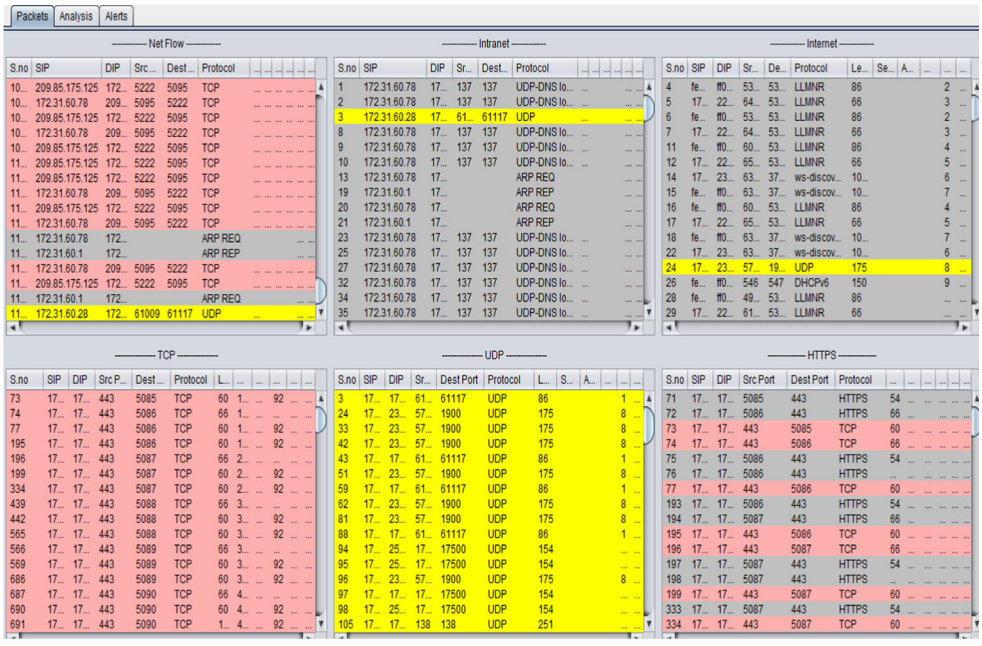


Fig. 5 Grid view. (*Source*: INTVS results)

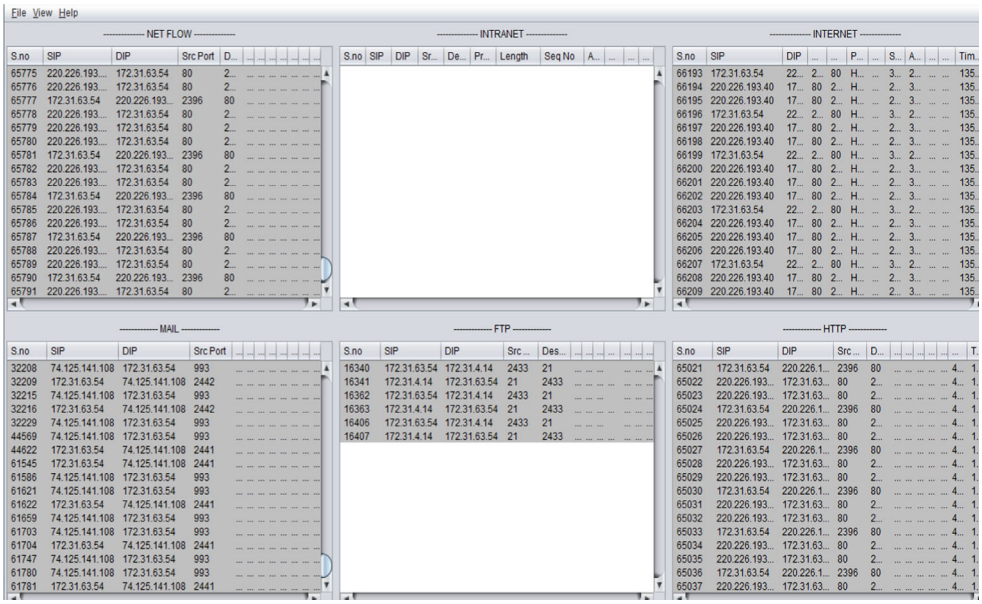


Fig. 6 Grid view of NetFlow, Intranet, Internet, Mail, FTP and HTTP traffic view. (*Source*: INTVS results)

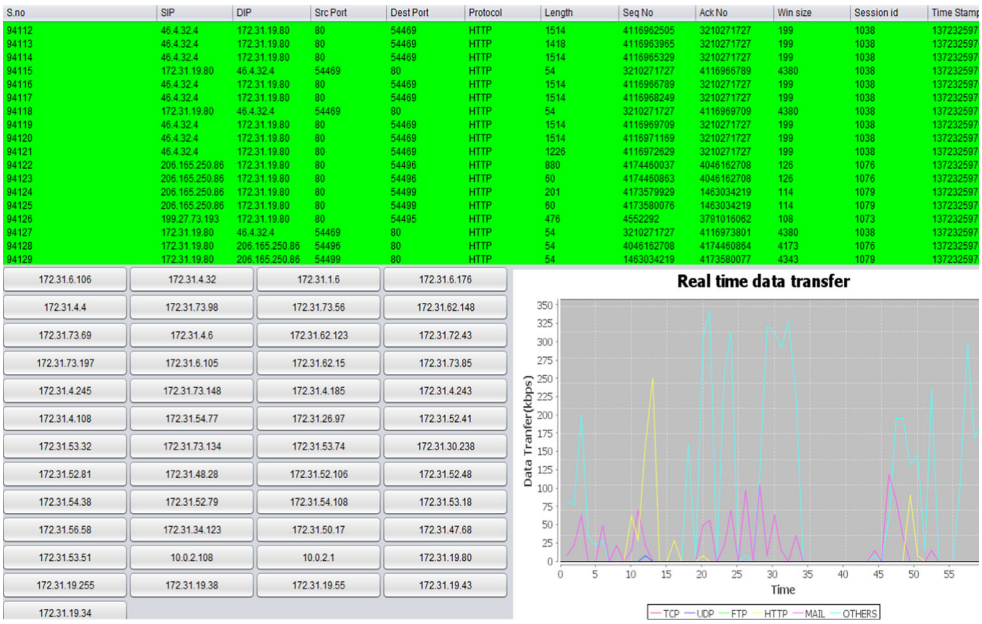


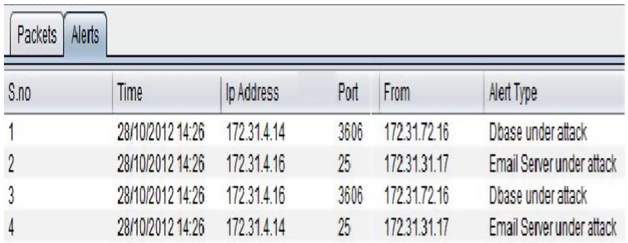
Fig. 7 View of HTTP/HTTPS Traffic. (*Source*: INTVS results)

5.4 Malicious activity detection module based on set theory

Algorithm 4: Malicious activity detection
Source: (Proposed by Authors)

1. BEGIN
2. A representing the set of ip address of legitimate user from inside of the network
3. B representing the set of authorized user from extranet
4. N representing Netflow
5. Malicious traffic = (A∪B)∩N
6. Using Total Resources= R
7. Resources for legitimate user from inside= R_i
8. Resources for legitimate user from outside=R_o
9. Restricted Resource to inside user=R_{ri}
10. Restricted Resource to outside user = R_{ro}
11. Policy for Restricted Resource to inside user= R_{ri_policy}
12. Policy for Restricted Resource to Outside user= R_{ro_policy}
13. Policy for Total Restricted Resource represents by RR= $R_{ri\_policy} \cup R_{ro\_policy}$
14. Total available resources can be represented by R= ($R_i \cup R_o$) ∪ ($R_{ri} \cup R_{ro}$)
15. Defining Sets for various user
16. Total Eligible user can be represented by U_rt
17. Legitimate User from inside= U_{ri}
18. Super User from inside= U_rs_i
19. Legitimate User from outside for extranet resources = U_{ro}
20. Guest user from public domain for web services only = U_r^g
21. Eligible users = U_rt
22. $U_rt = (U_{ri} \cup U_rs_i) \cup (U_{ro} \cup U_r^g)$
23. Total visitors of network = U^e
24. Un-authorized visitor denoted by $U_{Un-Auth} = U^e \cap U_rt$
25. Relationship between available resources and users $R \in U^e$
26. Authorized but Malicious user denoted by $U_{auth\_mal} = (U_{ri}\ R_{ri\_policy}) \cup (U_r\ o\ R_{ro\_policy})$
27. Total Malicious user $U_M = U_{Un-Auth} \cup U_{auth\_mal}$
28. C is representing set of color
29. C_{Mal} representing set of color for malicious user
30. C_{Auth} set of color for Non-Malicious user
31. $C_{Un-Auth}$ set of color for Un-authorized user
32. C_{UE} represents colors for each visitor of the network
33. $C = C_{Mal} \cup C_{Un-Auth}$
34. If user $U^e i \in U_M$ then
35. C_{UE} i =Get C_{Mal} for U_M i
36. Else
37. C_{UE} i =Get C_{Auth} for $U^e i$
38. M_{vis} will now plot $U^e i$ using C_{UE} i
39. END If
40. END

It is clearly evident that based on abovementioned network set theory algorithms, under any circumstances no visitors can infringe the network policy rules. Once unauthorised visitor tries to attempt to break and join the network, an alert will be generated momentarily if attempt is defined out of the network security policy terms and conditions. As a resultant packets and alerts details with time, IP address, Ports etc. can easily been identified as shown in Fig. 8.



S.no	Time	Ip Address	Port	From	Alert Type
1	28/10/2012 14:26	172.31.4.14	3606	172.31.72.16	Dbase under attack
2	28/10/2012 14:26	172.31.4.16	25	172.31.31.17	Email Server under attack
3	28/10/2012 14:26	172.31.4.16	3606	172.31.72.16	Dbase under attack
4	28/10/2012 14:26	172.31.4.14	25	172.31.31.17	Email Server under attack

Fig. 8 Alert analysis. (*Source*: INTVS results)

5.5 Algorithms and implementation of Platter view

This part of segment deals with the designing and implementation of Platter visualization engine.

Algorithm 5a: Platter view
Source: (Proposed by Authors)

1. BEGIN
2. data[][][] is a 3D array in which
3. data[i] represents a sub-net,
4. data[i][j] represents a machine n in data[i] subnet,
5. data[i][j][0] represents ip address if machine data[i][j]
6. data[i][j][1] represents total upload data of machine[i][j]
7. data[i][j][2] represents total download data of machine[i][j]
8. total_network_data will be used to hold total data transferred in the network
9. subnet_data[] is 1D array in which data[i] stores the total data transferred in subnet data[i]
10. screen[] holds the size of screen, in which screen[0] is height, and, screen[1] is width
11. origin[] holds the starting center holds the center of screen
12. sortNetwork(data, &total_network_data, subnet_data)
13. step_size:= (screen[0]-100)/(2 * length(data))
14. i:=0
15. For i<length(data) do
16. outer_circles[radius]:= step_size*(i+1)
17. j:=0
18. For j<length(data[i]) do
19. inner_circles[center_x] := center[x] + (outer_circle[i] * cos((ip.split('.')[3]*1.4)))
20. inner_circles[center_y] := center[y] + (outer_circle[i] * cos((ip.split('.')[3]*1.4)))
21. inner_circles[radius] := step_size/2
22. j++
23. End For
24. i++
25. End For
26. drawCircles (outer_circles, inner_circles, total_network_data, subnet_data, data)
27. END

Algorithm 5b: Sorting and swapping of circles according to number of machines in a subnet.
Source: (Proposed by Authors)

```
1. BEGIN
2. SET i:=0
3. For i<length(data) do
4. j:= 0
5. For j<length(data[i]) do
6. total_network_data:= total_network_data +
   data[i][j][1] + data[i][j][2]
7. subnet_data[i]: = subnet_data[i] + data[i][j][1] +
   data[i][j][2]
8. j++
9. End For
10. i++
11. End For
12. i:=length(data)
13. For i>0 do
14. SET j=1
15. For j<I do
16. If(subnet_data[j]<subnet_data[j-1]) then
17. swap(data[j],data[j-1])
18. End If
19. j++
20. End For
21. i++
22. End For
23. END
```

Algorithm 5c: Screen size responsive visualization *Source: (Proposed by Authors)*

```
1.  BEGIN
2.  SET i:=0
3.  For i<length(outer_circles) do
4.  setcolor(abs(118-(3*i),75,146))
5.  drawcircle(outer_circles[i])
6.  i++
7.  End For
8.  SET counter:=0
9.   SET prev_count:=0
10.  SET i:=0
11.  For i<length(inner_circles) do
12.  If(i<length(data[counter])) then
13.  temp:=data[counter][i-
      prev_count][1]+data[counter][i-prev_count][2]
14.  Else
15.  counter++
16.  prev_count=i,
17.  temp:= data[counter][i-
      prev_count][1]+data[counter][i-prev_count][2]
18.  end If
19.  If ((temp > total_network_data/length(data)) then
20.  setcolor(red) else if(temp > avg(subnet_data)) then
21.  setcolor(orange) else setcolor(green)
22.  drawcircle(inner_circle[i])
23.  end If
24.  end For
25.  END
```

The outcome after implementing these algorithms gives us global view of campus area wireless network on a single screen with the help of Platter analysis as shown in the Figs. 9 and 10. In Platter view analysis, circles are used to represent networks and rings are used to represent the machines. The smallest inner most circle in Platter analysis is representing the smallest network, and outer most biggest circle is representing the biggest network of a campus. A number of networks are detected as well as their respective machines are detected from captured data. These machines in a particular circle are represented by dots instead of rings as shown in the Fig. 10. The Platter analysis of this study is capable to display class B networks. The mouse hovering over the circle shows IP address of the network as well as number of live machines in that network and when mouse is hovered over a dot, it shows detail load (TCP, UDP, HTTP, FTP, downloading, uploading) on that machine.

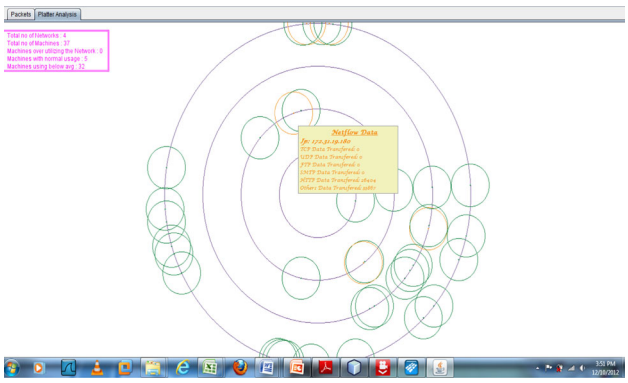


Fig. 9 Global view of CAN through Platter analysis. (*Source*: INTVS Results)

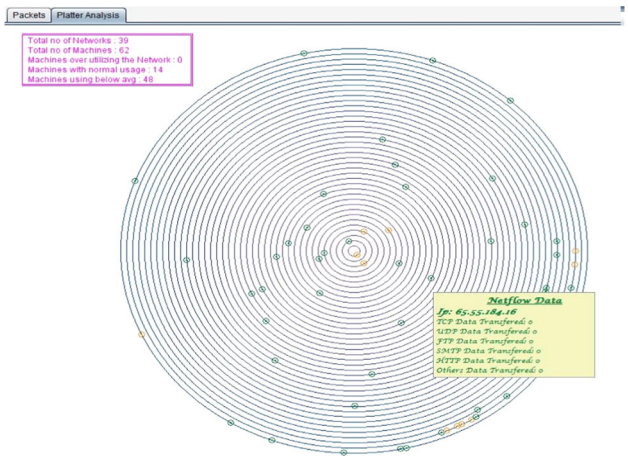


Fig. 10 Global view of CAN. (*Source*: INTVS results)

The red dot in a circle depicts that the data transfer from/to this machine is above average of the network bandwidth and blue dot depicts a normal bandwidth usage by that machine in a particular network. Nevertheless, on the similar lines, the color of circle also changes if the bandwidth consumed by the network is more and/or less than the average bandwidth allocated by the administrator focused.

5.6 Network and machine-level view

The details of machines are available and viewed using data filtering option as outlined in graphical representation in Fig. 11 (bottom left quadrant) through two-dimensional analysis approach. Moreover, this provides information about a machine consuming maximum/minimum bandwidth as per load laid down in a particular network using INTVS. In continuing with two-dimensional network analysis, for machine-level view in Fig. 11 (bottom right quadrant) displayed the various types of OSI Application Layer data traffic on the machine linked to the specified network.

5.7 Parallel coordinated analysis

Present section dealt with a particular condition in which an administrator finds a overloaded machine and can easily handled view related to connectivity of a machine with other network machines with the help of parallel coordinated Netmap analysis. In addition, details of its

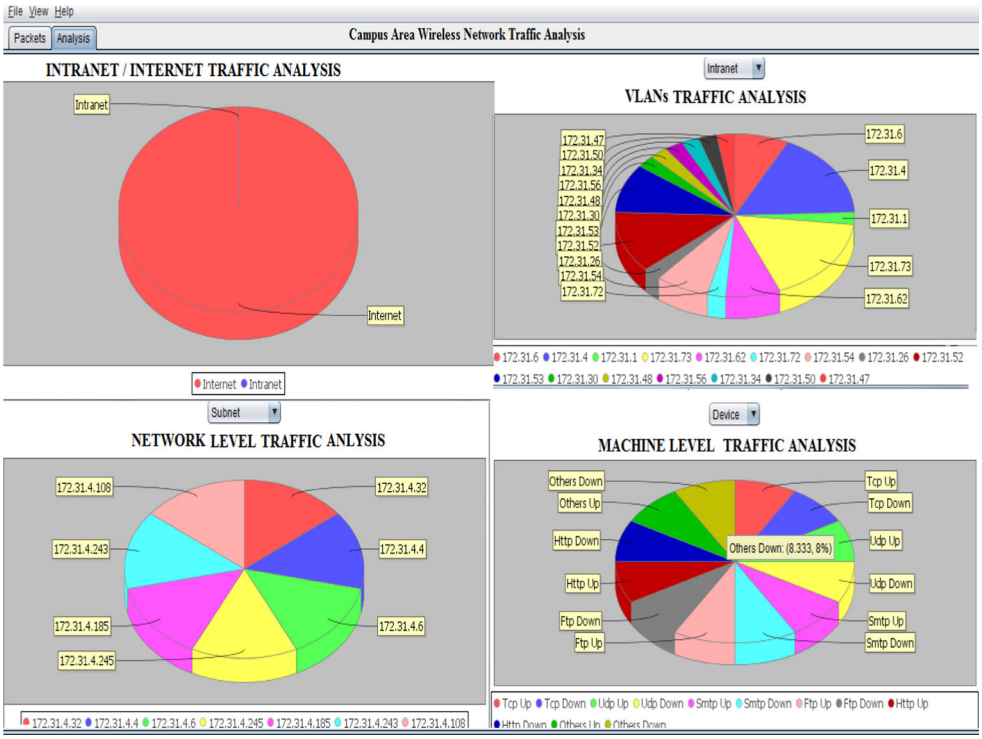


Fig. 11 Network and machine-level traffic analysis through two-dimensional analysis. (*Source*: INTVS results)

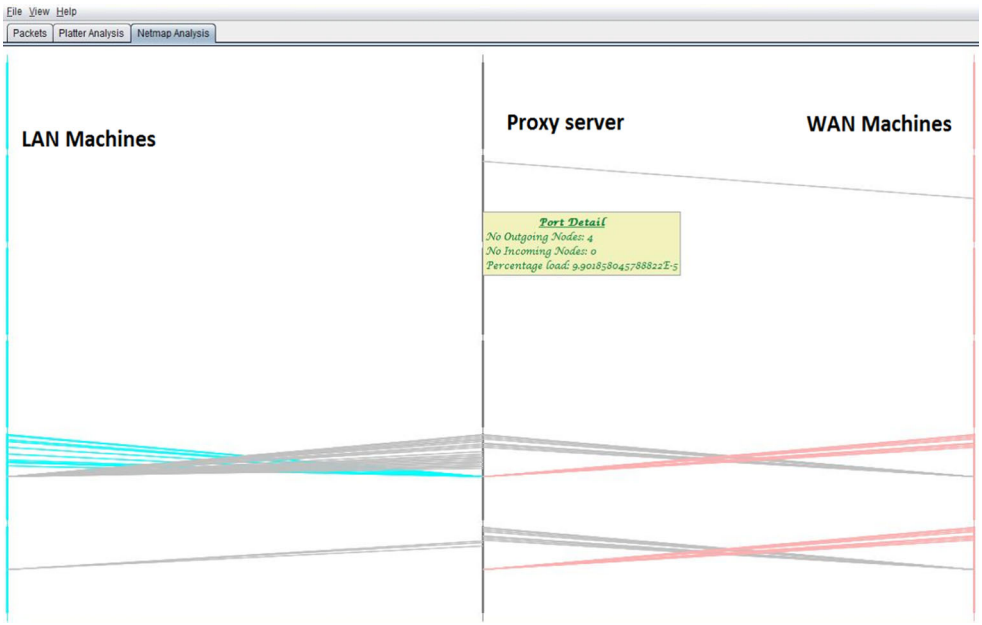


Fig. 12 Netmap analysis. (*Source*: INTVS results)

connectivity with other connected devices/machines can also be viewed on click of mouse-over facility available as shown in Fig. 12.

6 Conclusions

In digital and multitasking environment, it is highly important to manage the security of different type of networks in an around. This research study signifies and provided a new framework named as—INTVS framework, which primarily based on data mining techniques, utility and effective mechanisms. INTVS can capture raw packet, tokenized them and parse the data for analysis and visualization seamlessly in an integrated manner.

The INTVS can also classify and display the application layer data in real-time domain through Grid view scheme. The Platter view further supports and can display complete campus area wireless network traffic on a single screen for quick view, while detecting and reporting the security threats based on data mining techniques effectively. On the other hand, the drill-down facility of INTVS helps the network administrator to expertise which web service is most occupied and available load in a particular application. In conclusion, during different web services, INTVS helps the administrators to understand the loading unloading and scope to regulate the delayed response under various attacks to enable secure networks.

In future, people working in area of data mining and networks can easily adopt and implement INTVS framework significantly along with advanced approaches in the domain and further can be utilize for cloud-based services to critically understand and monitor the cost-effective solution under cumbersome situations.

Acknowledgments We are thankful CITM Department of Thapar University, Patiala, India for allowing the testing of INTVS.

References

1. Ware C (2012) Information visualization, perception for design (interactive technologies), 3rd edn
2. Lakkaraju K, Yurcik W, Lee A J (2004) NVisionIP: netflow visualizations of system state for security situational awareness. In: ACM workshop on visualization and data mining for computer security, VizSEC/DMSEC'04. ACM, pp 65–72
3. Ball R, Fink GA, North C (2004) Home-centric visualization of network traffic for security administration. In: ACM workshop on visualization and data mining for computer security, VizSEC/DMSEC'04. ACM, pp 55–64
4. Ahmad I, Abdullah AB, Alghamdi AS (2009) Application of artificial neural network in detection of probing attacks. In: IEEE symposium on industrial electronics and applications ISIEA 2009. IEEE, pp 557–562
5. Westphal C (2009) Data mining for intelligence, fraud, and criminal detection. CRC Press, Boca Raton. ISBN 13:978-1-4200-6723-1
6. Golnabi K, Min RK, Khan L, Al-Shaer E (2006) Analysis of firewall policy rules using data mining techniques. In: 10th IEEE/IFIP, network operations and management symposium, NOMS'2006. IEEE, pp 305–315
7. Vaarandi R (2009) Real-time classification of IDS alerts with data mining techniques. In: Military communications conference, MILCOM 2009. IEEE, pp 1–7
8. Swing E (1998) Flodar: flow visualization of network traffic. Comput Graph Appl IEEE 18(5):6–8
9. Estrin D, Handley M, Heidermann J, McCanne S, Xu Y, Yu H (2000) Network visualization with Nam, the VINT network administrator. IEEE Comput
10. Yin X, Yurcik W, Treaster M (2004) VisFlowConnect: NetFlow visualizations of link relationships for security situational awareness. In: ACM workshop on visualization and data mining for computer security,VizSEC/DMSEC'04. ACM. doi:1-58113-974-8/04/0010
11. Fink GA, Muessig P, North C (2005) Visual correlation of host processes and network traffic. In: IEEE workshop on visualization for computer security, VizSEC 05. IEEE, pp 11–19
12. Kim SS, Reddy ALN (2005) NetViewer: a network traffic and analysis tool. In: 19th large installation system administration conference, LISA'05(19). USENIX, pp 185–196
13. Estan C, Magin G (2005) Interactive traffic analysis and visualization with Wisconsin Netpy. In: 19th large installation system administration conference, LISA 05(19). USENIX, pp 177–184
14. Abdullah K, Lee CP, Conti G, Copeland JA, Stasko J (2005) IDS RainStorm: visualizing IDS alarms. In: IEEE workshop on visualization for computer security, VizSEC 05, pp 1–10
15. Conti G (2006) http://www.rumint.org. Accessed 20 Jan 2013
16. Marty R (2005) http://afterglow.sourceforge.net/. Accessed 20 Jan 2013
17. Marty R (2008) http://www.secviz.org/node/89. Accessed 20 Jan 2013
18. Reil JPV, Irwin B (2006) InetVis, a visual tool for network telescope traffic analysis. In: International conference on computer graphics, virtual reality, visualisation and interaction in Africa, AFRIGRAPH 2006. ACM, pp 85–89
19. Oberheide J, Goff M, Karir M (2006) Flamingo: visualizing internet traffic. In: Proceedings of the 10th IEEE/IFIP network operations and management symposium. IEEE, pp 150–161
20. Decker E, Hill S, Hebel K (2005) http://nfsen.sourceforge.net/#mozTocId201388. Accessed 20 Jan 2013
21. Godinho I, Meiguins B, Gonçalves A, Carmo C, Garcia M, Almeida L, Lourenço R (2007) PRISMA—a multidimensional information visualization tool using multiple coordinated views. In: 11th international conference on information visualization (IV'07). IEEE, pp 23–32
22. Taylor T, Paterson D, Glanfield J, Gates C, Brooks S, McHugh J (2009) FloVis: flow visualization system. In: Cybersecurity applications and technology conference for homeland security. IEEE, pp 186–198
23. Allen M, McLachlan P (2009) NAV—network analysis visualization. University of British Columbia. [Online, 29 May 2009]
24. Goodall JR, Sowul M (2009) VIAssist: visual analytics for cyber defense. In: Technologies for homeland security, HST'09. IEEE, pp 143–150
25. Jiawan Z, Liang L, Liangfu L, Ning Z (2008) A novel visualization approach for efficient network scans detection. In: International conference on security technology, SECTECH'08. IEEE, pp 23–26
26. Osborne G, Turnbull B, Slay J (2010) The 'Explore, Investigate and Correlate' (EIC) conceptual framework for digital forensics information visualisation. In: ARES'10 international conference on availability, reliability, and security. IEEE, pp 629–634

27. Lu LF, Zhang JW, Huang ML, Fu L (2010) A new concentric-circle visualization of multi-dimensional data and its application in network security. J Vis Lang Comput 21:194–208
28. Creese S, Goldsmith M, Moffat N, Happa J, Agrafiotis I (2013) CyberVis: visualizing the potential impact of cyber attacks on the wider enterprise. In: International conference on technologies for homeland security, HST'2013. IEEE, pp 73–79
29. Singh MP, Subramanian N, Rajamenakshi (2009) Visualization of Flow Data Based on Clustering Technique for Identifying Network Anomalies. In: IEEE symposium on industrial electronics and applications, ISIEA 2009. IEEE, pp 973–978
30. Shneiderman B (1996) The eyes have it: a task by data type taxonomy of information visualizations. In: IEEE symposium on visual languages. IEEE, pp 336–343