Project 1

For due date see learn.bcit.ca

Network Intrusion Detection Classification

Deliverables

- The prepared report (in `.pdf` format) following the guidelines in `comp8085_report.tex`.
- The final `NIDS.py` which contains the implementation of your feature selection (and potentially data visualization) code as well as the code for the experiments of the two classification parts. `NIDS.py` should provide the functionality to load the trained models and test them without going through training.
- A compressed file (if necessary; see the comments in Parts 2 and 3) containing your pickled trained models so that they can be placed besides `NIDS.py` to run the inference code.

Background

This project description provides the data and requirement for you to create a network intrusion detection agent. To get familiar with the topic, first, lets read the following text from Moustafa et al. [1]:

> Currently, due to the massive growth in computer networks and applications, many challenges arise for cyber security systems. Intrusions/attacks can be defined as a set of events which are able to compromise the principles of computer systems, e.g. availability, authority, confidentiality and integrity. Firewall systems cannot detect modern attack environments and are not able to analyse network packets in depth. Because of these reasons, Intrusions Detection Systems are designed to achieve high protection for the cyber security infrastructure.
>
> A Network Intrusion Detection System (NIDS) monitors network traffic flow to identify attacks. NIDSs are classified into misuse/signature and anomaly based. The signature-based matches the existing of known attacks to detect intrusions. However, in the anomaly-based, a normal profile is created from the normal behavior of the network, and any deviation from this is considered as attack. Further, the signature-based NIDSs cannot detect unknown attacks, and for these anomaly NIDS are recommended in many studies.
>
> The effectiveness of NIDS is evaluated based on their performance to identify attacks which requires a comprehensive data set that contains normal and abnormal behaviors.

Project 1

For due date see learn.bcit.ca

Researchers at the University of New South Wales (UNSW; Australia) have collected 100 GB of the raw network traffic using the IXIA PerfectStorm tool and created *UNSW-NB15* dataset containing 49 features with the class label indicating whether the record is normal or not (Table 1 provides the full list of the extracted features). The dataset is *a hybrid of real modern normal activities and synthetic contemporary attack behaviours*, and has nine types of attacks, namely, **Fuzzers**, **Analysis**, **Backdoors**, **DoS**, **Exploits**, **Generic**, **Reconnaissance**, **Shellcode** and **Worms**. You are highly recommended to read the full paper explaining the data and how it was generated to get more familiar with it (a copy of the paper is provided to you with this file).

Since the full dataset is quite big (and unbalanced), you work on a subset of the dataset which is balanced and smaller. The `UNSW-NB15-BALANCED-TRAIN.csv` data file that you receive follows the same composition rules as the full data except that it has 449,797 training records.

Project Description

Your task is to read about the data and create a `NIDS.py` script which receives the name of a test file (with the exact same format as `UNSW-NB15-BALANCED-TRAIN.csv`) as well as a classifier name (e.g. it should be callable with the command '`python NIDS.py <heldout_testset.csv> <selected_classification_method_name> <task> <optional_load_model_name>`'), classifies the network traffic records in it and produces the proper classification reports. In each task you will consider a separate prediction category; `Label` (predicting whether the record is normal or not) or `attack_cat` (predicting the attack category). **Clearly, you are not allowed to use `attack_cat` and `Label` attributes as input features to any of your models**. However, you will use those categories to select proper features in Part 1.

Part 1: Feature Analysis and Selection

The dataset provided with this project comes with 47 features/attributes (each in one of the columns except for `attack_cat` and `Label` attributes). Chances are that not all 47 of these features are as useful in predicting the target labels (`attack_cat` or `Label`). So you need to perform analysis and figure out which ones are the best and most relevant ones for your classification tasks. The selected feature sets could be different for each of `attack_cat` and `Label` attributes.

Hint: for categorical attributes like `proto`, you can use the `factorize` function from `pandas` library to convert them to numerical format. You may want to do this for each of such attributes before performing the data analysis part. As well, you may also convert `Object` formatted columns to string type using `df['column_name'].astype('str')`, and make sure you deal with missing values properly.

To do this part:

1. As you notice, you are not provided with a validation set. Reserve a chunk of the provided training data to perform the internal testing and the model selection and in the **Approach section** of your report explain how you did it and what is the data size of each part.

2. Choose **different analysis techniques** (you need to choose one per group member).

   - You can choose from different techniques such as feature co-variance/correlation analysis, recursive feature elimination, principal component analysis, entropy based feature importance analysis, ... Just make sure the analysis types are different.

3. In the **Approach section** of your report, explain how each technique works and how it can help in feature selection.

4. In the **Experiments section** of your report, provide possible charts/graphs that the feature analysis techniques output as well as potentially extracted scores/stats from each. Finalize your selected feature set for performing each classification task and in the report motivate why this is your final choice.
   Make sure you do not use the held-out validation and test sets for these analysis.

5. Take one of the classifiers (preferably the best scoring one) from the next section and compare its classification scores with and without feature selection applied to the data. You will need to do this once for `Label` and one other time for `attack_cat` prediction. Provide the results of this experiment in the **Experiments section** of your report.

6. For the analysis of the results of each technique in the **Experiments section** of your report, consider **at most** one page as the limit. Please note that your analysis does not have to reach the limit. This is just to make sure one does not spend too much space on this part. Also,

   - **Do not** throw in tens of graphs without any explanation, your explanation matters more to me than the graphs!
   - **Do not** put the analysis code in the report, just explain your findings and point me to the proper source code file/line in your submitted source code (if necessary).

Part 2: `Label` Classification

Make sure your model is not using your validation set as training data. The validation and test sets **must remain the same** in all of your experiments. Optionally you can also consider k-fold cross validation.

To do this part:

1. Choose **different classification techniques** (you need to choose one per group member, and you can simply choose the pre-implemented `sklearn` classifiers) and in the **Approach section** of your report explain how each classifier works.
   Make sure you don't mistake regression models for the classifiers.
   As well, `DecisionTree` and `RandomForest` are not of different classification types.
2. Train classifiers and using your held out validation set try to find the best settings for each. In your **Experiments section**, explain the process of finding these best settings as well as the `Label` classification scores for each classifier on the held-out test set.
   Your experimental results should contain three formatted tables looking like the following:

   Classifier: <classifier-name>

	precision	recall	f1-score	support
0	0.00	0.00	0.00	00000
1	0.00	0.00	0.00	00000
accuracy			0.00	00000
macro avg	0.00	0.00	0.00	00000
weighted avg	0.00	0.00	0.00	00000

3. For the analysis of the results of each classifier (**Experiments section**), report which classifier worked the best and explain why it worked better than the others.
4. If your classifiers train pretty quickly, you may have them train every-time I run the inference code (in this case you need to receive the train data in addition to the held-out test set as the input arguments to the program). Otherwise, pickle your trained models and submit them along with the code and the report. To do so, pick a proper method for saving the trained models and modify the code in a way that each model can be loaded and tested (on the held-out test set) without training. You may use `pickle` library for this.

- All three classifiers you submit must report F1 scores above 0.9. I will also test your submission with my held-out test set (which has a similar distribution to the data that you have) and expect your submission to get F1 scores above 0.85.

Part 3: `attack_cat` Classification

The main classification task in this report is concerned with classification of attack categories. As you might have noticed, the binary classification of `Label` attribute is quite an easy task and you can get high accuracy scores (why should this happen?). In this section, you will try

classifying the records which are labelled as an attack. Our objective is to get the **highest Macro-F1** classification score.

To do this part:

1. Like previous part choose **different classification techniques** (you need to choose one per group member), and in the **Approach section** of your report explain how each classifier works (if you have not already covered it in the previous task).
   Make sure you don't mistake regression models for the classifiers.
   As well, `DecisionTree` *and* `RandomForest` *are not of different classification types.*

   - You may choose from decision trees, perceptron classifiers, linear classifiers with logistic regression, kNNs, SVMs, and any other classifiers you like. The only important criteria is the accuracy threshold mentioned in the next page.

2. Train classifiers and using your held out validation set try to find the best settings for each. In your **Experiments section**, explain the process of finding these best settings as well as the `attack_cat` classification scores for each classifier on the held-out test set. Make sure you report both Micro-F1 and Macro-F1 scores[1]. Your experimental results should contain three formatted tables looking like the following:

   Classifier: <classifier-name>

	precision	recall	f1-score	support
None	0.00	0.00	0.00	00000
Generic	0.00	0.00	0.00	00000
Fuzzers	0.00	0.00	0.00	00000
Exploits	0.00	0.00	0.00	00000
DoS	0.00	0.00	0.00	00000
Reconnaissance	0.00	0.00	0.00	00000
Analysis	0.00	0.00	0.00	00000
Shellcode	0.00	0.00	0.00	00000
Backdoors	0.00	0.00	0.00	00000
Worms	0.00	0.00	0.00	00000
micro avg	0.00	0.00	0.00	00000
macro avg	0.00	0.00	0.00	00000
weighted avg	0.00	0.00	0.00	00000

3. For the analysis of the results of each classifier (**Experiments section**), report which classifier worked the best and explain why it worked better than the others. As well,

[1] For more info on the difference between the two please check out this [LINK].

explain what is the main issue in the dataset that the classifiers would suffer from and what was your approach to deal with it.

4. If your classifiers train pretty quickly, you may have them train every-time I run the inference code (in this case you need to receive the train data in addition to the held-out test set as the input arguments to the program). Otherwise, pickle your trained models and submit them along with the code and the report. To do so, pick a proper method for saving the trained models and modify the code in a way that each model can be loaded and tested (on the held-out test set) without training.

- All three classifiers you submit must report Macro-F1 scores above 0.45. I will also test your submission with my held-out test set and expect your submission to get Macro-F1 scores above 0.40. Micro-F1 should also be above 0.9 on the validation set and above 0.85 on my held-out test set.

A few tips to lessen the complexity of this part:

- If your results table has duplicate records (which should be the case!), consider normalization of attack labels in the train set, **but not in your validation and test sets**!
- Don't forget that in the provided train data file, `attack_cat` is **empty** for non-attack records!
- You may use the prediction result of your `Label` classifier to reduce the complexity of multi-class classification for `attack_cat` classifier.
- **Important**: there is no requirement that you train completely different classifiers for `Label` and `attack_cat` parts. For example, you may train a decision tree classifier for `Label` and another decision tree classifier for `attack_cat`. The only important criteria for each classification technique is that it reports F1 scores above 0.9 for `Label` and reports Macro-F1 scores above 0.45 for `attack_cat`.
  - Also, you may try three feature selection techniques for `Label` and three other ones for `attack_cat`, however, **this is not a necessity!** you can simply perform one feature selection step and use the same features (dataset columns) in both parts.

References

[1] Moustafa, N., & Slay, J. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In 2015 military communications and information systems conference (MilCIS) (pp. 1–6). Can be found under https://www.researchgate.net/profile/Nour-Moustafa/publication/

Appendix

Table 1: The description of the extracted features as explained in UNSW-NB15_features.csv file.

Name	Type	Description
srcip	nominal	Source IP address
sport	integer	Source port number
dstip	nominal	Destination IP address
dsport	integer	Destination port number
proto	nominal	Transaction protocol
state	nominal	Indicates to the state and its dependent protocol, e.g. ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, and (-) (if not used state)
dur	Float	Record total duration
sbytes	Integer	Source to destination transaction bytes
dbytes	Integer	Destination to source transaction bytes
sttl	Integer	Source to destination time to live value
dttl	Integer	Destination to source time to live value
sloss	Integer	Source packets retransmitted or dropped
dloss	Integer	Destination packets retransmitted or dropped
service	nominal	http, ftp, smtp, ssh, dns, ftp-data ,irc and (-) if not much used service
Sload	Float	Source bits per second
Dload	Float	Destination bits per second
Spkts	integer	Source to destination packet count
Dpkts	integer	Destination to source packet count
swin	integer	Source TCP window advertisement value
dwin	integer	Destination TCP window advertisement value
stcpb	integer	Source TCP base sequence number
dtcpb	integer	Destination TCP base sequence number
smeansz	integer	Mean of the flow packet size transmitted by the src
dmeansz	integer	Mean of the flow packet size transmitted by the dst
trans_depth	integer	Represents the pipelined depth into the connection of http request/response transaction
res_bdy_len	integer	Actual uncompressed content size of the data transferred from the server's http service.
Sjit	Float	Source jitter (mSec)
Djit	Float	Destination jitter (mSec)

Stime	t_stamp	record start time
Ltime	t_stamp	record last time
Sintpkt	Float	Source interpacket arrival time (mSec)
Dintpkt	Float	Destination interpacket arrival time (mSec)
tcprtt	Float	TCP connection setup round-trip time, the sum of 'synack' and 'ackdat'.
synack	Float	TCP connection setup time, the time between the SYN and the SYN_ACK packets.
ackdat	Float	TCP connection setup time, the time between the SYN_ACK and the ACK packets.
is_sm_ips_ports	Binary	If source (1) and destination (3)IP addresses equal and port numbers(2)(4) equal then, this variable takes value 1 else 0
ct_state_ttl	Integer	No. for each state (6) according to specific range of values for source/destination time to live (10) (11).
ct_flw_http_mthd	Integer	No. of flows that has methods such as Get and Post in http service.
is_ftp_login	Binary	If the ftp session is accessed by user and password then 1 else 0.
ct_ftp_cmd	integer	No of flows that has a command in ftp session.
ct_srv_src	integer	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
ct_srv_dst	integer	No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26).
ct_dst_ltm	integer	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
ct_src_ ltm	integer	No. of connections of the same source address (1) in 100 connections according to the last time (26).
ct_src_dport_ltm	integer	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
ct_dst_sport_ltm	integer	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
ct_dst_src_ltm	integer	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).

attack_cat	nominal	The name of each attack category. In this data set , nine categories e.g. Fuzzers, Analysis, Backdoors, DoS Exploits, Generic, Reconnaissance, Shellcode and Worms
Label	binary	0 for normal and 1 for attack records

UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems

(UNSW-NB15 Network Data Set)

Nour Moustafa, IEEE student Member, Jill Slay
School of Engineering and Information Technology
University of New South Wales at the Australian Defence Force Academy
Canberra, Australia
E-mail: nour.abdelhameed@student, j.slay@{.adfa.edu.au}

Abstract— **One of the major research challenges in this field is the unavailability of a comprehensive network based data set which can reflect modern network traffic scenarios, vast varieties of low footprint intrusions and depth structured information about the network traffic. Evaluating network intrusion detection systems research efforts, KDD98, KDDCUP99 and NSLKDD benchmark data sets were generated a decade ago. However, numerous current studies showed that for the current network threat environment, these data sets do not inclusively reflect network traffic and modern low footprint attacks. Countering the unavailability of network benchmark data set challenges, this paper examines a UNSW-NB15 data set creation. This data set has a hybrid of the real modern normal and the contemporary synthesized attack activities of the network traffic. Existing and novel methods are utilised to generate the features of the UNSW-NB15 data set. This data set is available for research purposes and can be accessed from the link[1].**

Keywords- UNSW-NB15 data set; NIDS; low footprint attacks; pcap files; testbed

I. INTRODUCTION

Currently, due to the massive growth in computer networks and applications, many challenges arise for cyber security research. Intrusions /attacks can be defined as a set of events which are able to compromise the principles of computer systems, e.g. availability, authority, confidentiality and integrity [1]. Firewall systems cannot detect modern attack environments and are not able to analyse network packets in depth. Because of these reasons, IDSs are designed to achieve high protection for the cyber security infrastructure [2].

A Network Intrusion Detection System (NIDS) monitors network traffic flow to identify attacks. NIDSs are classified into misuse/signature and anomaly based [4]. The signature based matches the existing of known attacks to detect intrusions. However, in the anomaly based, a normal profile is created from the normal behavior of the network, and any deviation from this is considered as attack [3] [4]. Further, the signature based NIDSs cannot detect unknown attacks, and for these anomaly NIDS are recommended in many studies [4] [5].

The effectiveness of NIDS is evaluated based on their performance to identify attacks which requires a comprehensive data set that contains normal and abnormal behaviors [6]. Older benchmark data sets are KDDCUP 99 [7] and NSLKDD [8] which have been widely adopted for evaluating NIDS performance. It is perceived through several studies [6][9][10][11], evaluating a NIDS using these data sets does not reflect realistic output performance due to several reasons. First reason is the KDDCUP 99 data set contains a tremendous number of redundant records in the training set. The redundant records affect the results of detection biases toward the frequent records [10]. Second, there are also multiple missing records that are a factor in changing the nature of the data [9]. Third, The NSLKDD data set is the improved version of the KDDCUP 99, it tackles the several issues such as data unbalancing among the normal/abnormal records and the missing values [12]. However, this data set is not a comprehensive representation of a modern low foot print attack environment.

The above reasons have instigated a serious challenge for the cyber security research group at the Australian Centre for Cyber Security (ACCS)[2] and other researchers of this domain around the globe. Countering this challenge, this paper provides an effort in creating a UNSW-NB15 data set to evaluate NIDSs. The IXIA PerfectStorm tool[3] is utilised in the Cyber Range Lab of the ACCS to create a hybrid of the modern normal and abnormal network traffic. The abnormal traffic through the IXIA tool simulates nine families of attacks that are listed in Table VIII. The IXIA tool contains all information about new attacks that are updated continuously from a CVE site[4]. This site is a dictionary of publicly known information security vulnerabilities and exposures. Capturing network traffic in the form of packets, the tcpdump[5] tool is used. The simulation period was 16 hours on Jan 22, 2015 and 15 hours on Feb 17, 2015 for capturing 100 GBs. Further, each pcap file is divided into 1000 MB using the tcpdump tool. Creating reliable features from the pcap files, Argus[6] and

[1] http://www.cybersecurity.unsw.adfa.edu.au/ADFA%20NB15%20Datasets/.

[2] http://www.accs.unsw.adfa.edu.au/
[3] http://www.ixiacom.com/products/perfectstorm
[4] https://cve.mitre.org/
[5] http://www.tcpdump.org/
[6] http://qosient.com/argus/index.shtml

Bro-IDS[7] tools are utilised. Additionally, twelve algorithms are developed using a C# language to analyse in-depth the flows of the connection packets. The data set is labelled from a ground truth table that contains all simulated attack types. This table is designed from an IXIA report that is generated during the simulation period. The key characteristics of the UNSW-NB15 data set are a hybrid of the real modern normal behaviors and the synthetical attack activities.

The rest of the paper is organised as follows: section 2 examines the general goal and orientation of any IDS data set. Section 3 exposes in-detail the existing benchmark datasets shortcomings. The synthetic environment configuration and generation of UNSW-NB15 details are given in section 4. Section 5 is a comparative analysis between the KDDCUP99 and the UNSW-NB15 data set. Section 6 displays the final shape about the files of the UNSW-NB15 data set. Finally, section 7 concludes the work and future intentions.

II. THE GOAL AND ORIENTATION OF A NIDS DATA SET

A NIDS data set can be conceptualized as relational data [6]. Input to a NIDS is a set of data records. Each record consists of attributes of different data types (e.g., binary, float, nominal and integer) [6]. The label assigns each record of the data, either normal is 0 or abnormal is 1. Labelling is done by matching processed record, according to the particular NIDS scenario with the ground truth table of all transaction records.

III. CRITICISMS OF EXITING DATA SETS

A quality of the NIDS data set reflects two important characteristics are a comprehensive reflection of contemporary threat and inclusive normal range of traffic. The quality of the data set ultimately affects the reliable outcome of any NIDS [6] [9]. In this section the disadvantages of existing data sets for NIDS are explored in the perspective of data set quality. The most widely adopted data sets for NIDS are KDDCUP99, and its improved version NSL-KDD.

A. KDDCup99 Data Set

Generating DARPA98 [13], (IST) group of Lincoln laboratories at MIT University performed a simulation with normal and abnormal traffic in a military network (U.S. Air Force LAN) environment. The simulation ended with nine weeks of raw tcpdump files. The training data size was about four GBs and consisted of compressed binary tcpdump files from seven weeks of network traffic. This was processed into approximately five million connection records. The simulation provided two weeks of test data which contained two million connection records [7] [13].

Upgrading DARAP98 network data features comprehensiveness, utilising the same environment (U.S. Air Force LAN), the simulation ended with 41 features for each connection along with the class label using Bro-IDS tool. The upgraded version of DARAP98 is referred to as KDDCUP99. In the KDDCUP99 data set, the whole extracted features were divided into three groups of intrinsic features, content features and traffic features. Further, attack records in this data set are categorised into four vectors (e.g., DoS, Probe, U2R, and R2L). The training set of KDDCUP99 included 22 attack types and test data contained 15 attack types [13] [7].

A number of IDS researchers as have utilised these datasets due to their public availability. However, many researchers have reported majorly three important disadvantages of these datasets [6] [9] [10] [11] [12] which can affect the transparency of the IDS evaluation. First, every attack data packets have a time to live value (TTL) of 126 or 253, whereas the packets of the traffic mostly have a TTL of 127 or 254. However, TTL values 126 and 253 do not occur in the training records of the attack [9]. Second, the probability distribution of the testing set is different from the probability distribution of the training set, because of adding new attack records in the testing set [10][12]. This leads to skew or bias classification methods to be toward some records rather than the balancing between the types of attack and normal observations. Third, the data set is not a comprehensive representation of recently reported low foot print attack projections [11].

B. NSLKDD Data Set

According to [12] considering the three goals, an upgraded version of the KDD data set was created and it is referred to as NSLKDD. The first goal was, removing the duplication of the record in the training and test sets of the KDDCUP99 data set for the purpose of eliminating classifiers biased to more repeated records. Secondly, selecting a variety of the records from different parts of the original KDD data set is to achieve reliable results from classifier systems. Third, eliminating the unbalancing problem among the number of records in the training and testing phase is to decrease the False Alarm Rates (FARs). The major disadvantage of NSLKDD is that, it does not represent the modern low foot print attack scenarios [9] [12].

IV. UNSW-NB15 DATA SET

In this section, the synthetic environment configuration and generation of UNSW-NB15 details are presented. The section includes mainly the testbed configuration details and the whole processes which involved in generating UNSW-NB15 from the configured testbed.

A. An IXIA tool Testbed Configuration

According to Fig. 1, the IXIA traffic generator is configured with the three virtual servers. The servers 1 and 3 are configured for normal spread of the traffic while server 2 formed the abnormal/malicious activities in the network traffic. Establishing the intercommunication between the servers, acquiring public and private network traffic, there are two virtual interfaces having IP addresses, *10.40.85.30* and *10.40.184.30.* The servers are connected to hosts via two routers. The router 1 has *10.40.85.1* and *10.40.182.1* IP addresses, whereas router 2 is configured with *10.40.184.1* and

[7] https://www.bro.org/index.html

10.40.183.1 IP addresses. These routers are connected to the firewall device that is configured to pass all the traffic either normal or abnormal. The tcpdump tool is installed on the router 1 to capture the Pcap files of the simulation uptime. Moreover, the central intent of this whole testbed was to capture the normal or abnormal traffic, which was originated from the IXIA tool and dispersed among network nodes (e.g., servers and clients). Importantly, the IXIA tool is utilised as an attack traffic generator along with as normal traffic, the attack behaviour is nourished from the CVE site for the purpose of a real representation of a modern threat environment.

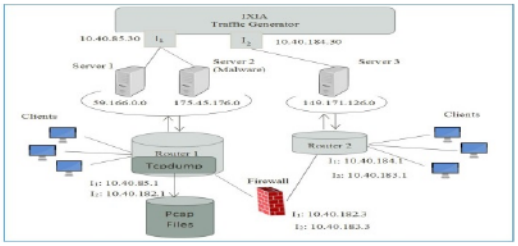

Figure 1. The Testbed Visualization for UNSW-NB15

Due to the speed of network traffic and the way of exploiting by modern attacks, the IXIA tool is configured to generate one attack per second during the first simulation to capture the first 50 GBs. On the other hand, the second simulation is configured to make ten attacks per second to extract another 50 GBs.

B. *Traffic Analysis*

The traffic analysis is described for the cumulative flows during the period of the simulation while generating the UNSW-NB15 data set. In Table I, the data set statistics are provided which represents the simulation period, the flows numbers, the total of source bytes, the destination bytes, the number of source packets, the number of destination packets, protocol types, the number of normal and abnormal records and the number of unique source/destination IP addresses.

TABLE I. DATA SET STATISTICS

Statistical features		**16 hours**	**15 hours**
No._of_flows		987,627	976,882
Src_bytes		4,860,168,866	5,940,523,728
Des_bytes		44,743,560,943	44,303,195,509
Src_Pkts		41,168,425	41,129,810
Dst_pkts		53,402,915	52,585,462
Protocol types	TCP	771,488	720,665
	UDP	301,528	688,616
	ICMP	150	374
	Others	150	374
Label	Normal	1,064,987	1,153,774
	Attack	22,215	299,068
Unique	Src_ip	40	41
	Dst_ip	44	45

In Fig. 2, the concurrent transactions with respect the time which are presented during the 16 hours of the simulation on Jan 22, 2015 and the 15 hours of Feb 17, 2015. The x-axis shows the time of each 10 seconds and the y-axis represents the number of Kbytes that is sniffed during each simulation period.

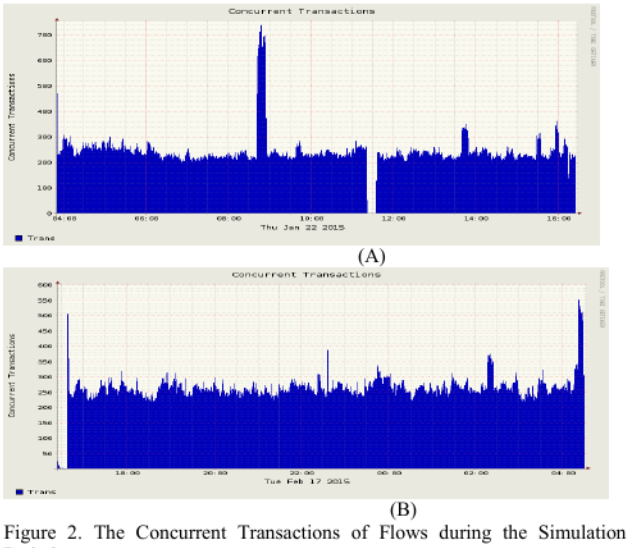

Figure 2. The Concurrent Transactions of Flows during the Simulation Periods.

C. *Architectural Framework*

The whole architecture which is involved in generating the final shape of the UNSW-NB15 from pcap files to CSV files with 49 features (attributes in any CSV file) is presented in Fig. 3. All the 49 features of the UNSW-NB15 data set are elaborated from Tables II-VII along with the generation sequence explanation for understanding convenience.

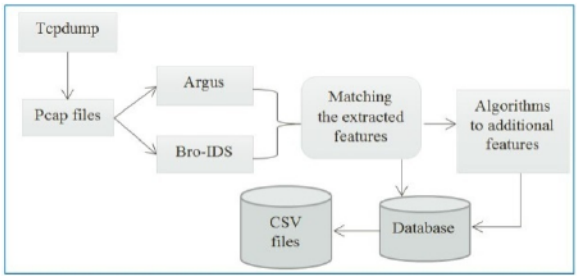

Figure 3. Framework Architecture for Generating UNSW-NB15 data set

When the simulation was running on the testbed presented in Fig. 1, the pcap files are generated by using the tcpdump tool. The features of the UNSW-NB15 data set are extracted by using Argus, Bro-IDS tools and twelve algorithms are developed using c# programming language as shown in Fig. 3. Moreover, these features are matched according to the equal flow features as listed in Table II. These tools are installed and are configured on Linux Ubuntu 14.0.4. The detailed formatting description of the UNSW-NB15 data set is elaborated in the following sections.

D. *The extracted features from the Argus and Bro-IDS Tools*

Argus tool processes raw network packets (e.g., pcap files) and generates attributes/features of the network flow packets. The Argus tool consists of an Argus-server and Argus-clients.

The Argus-server writes pcap files of receiving packets in Argus files in the binary format. The Argus clients extract the features from the Argus files.

Bro-IDS tool is an open-source network traffic analyser. It is predominantly a security monitor that inspects all network traffic against malicious activities. The Bro-IDS tool is configured to generate three log files from the pcap files. First, the conn file records all connection information seen on the pcap files. Second, the http file includes all HTTP requests and replies. Third, the ftp file records all activities of a FTP service.

Finally, the output files of the two different tools, Argus and Bro-IDS are stored in the SQL Server 2008[8] database to match the Argus and Bro-IDS generated features by using the flow features as reflected in Table II.

TABLE II. FLOW FEATURES

#	Name	T.	Description
1	*srcip*	N	Source IP address
2	*sport*	I	Source port number
3	*dstip*	N	Destination IP address
4	*dsport*	I	Destination port number
5	*proto*	N	Transaction protocol

E. The matched features of the Argus and Bro-IDS Tools

These features include a variety of packet-based features and flow-based features. The packet based features assist the examination of the payload beside the headers of the packets. On the contrary, for the flow based features and maintaining low computational analysis instead of observing all the packets going through a network link, only connected packets of the network traffic are considered. Moreover, the flow-based features are based on a direction, an inter-arrival time and an inter-packet length [6] (mentioned in Tables III and IV, as well as they are executed in the connection features of Table VI).The matched features are categorised into three groups: Basic, Content, and Time which were described in Tables III, IV and V, respectively.

TABLE III. BASIC FEATURES

#	Name	T	Description
6	*state*	N	The state and its dependent protocol, e.g. ACC, CLO, else (-)
7	*dur*	F	Record total duration
8	*sbytes*	I	Source to destination bytes
9	*dbytes*	I	Destination to source bytes
10	*sttl*	I	Source to destination time to live
11	*dttl*	I	Destination to source time to live
12	*sloss*	I	Source packets retransmitted or dropped
13	*dloss*	I	Destination packets retransmitted or dropped
14	*service*	N	http, ftp, ssh, dns ..,else (-)
15	*sload*	F	Source bits per second
16	*dload*	F	Destination bits per second
17	*spkts*	I	Source to destination packet count
18	*dpkts*	I	Destination to source packet count

[8] http://www.microsoft.com/en-au/download/details.aspx?id=26113

Importantly, the features from 1-35 represent the integrated gathered information from data packets. The majority of features are generated from header packets as reflected in Tables II-V. It is acknowledged that the UNSW-NB15 data set creates additional flow based features as described in the following section.

TABLE IV. CONTENT FEATURES

#	Name	T	Description
19	*swin*	I	Source TCP window advertisement
20	*dwin*	I	Destination TCP window advertisement
21	*stcpb*	I	Source TCP sequence number
22	*dtcpb*	I	Destination TCP sequence number
23	*smeansz*	I	Mean of the flow packet size transmitted by the src
24	*dmeansz*	I	Mean of the flow packet size transmitted by the dst
25	*trans_depth*	I	the depth into the connection of http request/response transaction
26	*res_bdy_len*	I	The content size of the data transferred from the server's http service.

TABLE V. TIME FEATURES

#	Name	T	Description
27	*sjit*	F	Source jitter (mSec)
28	*djit*	F	Destination jitter (mSec)
29	*stime*	T	record start time
30	*ltime*	T	record last time
31	*sintpkt*	F	Source inter-packet arrival time (mSec)
32	*dintpkt*	F	Destination inter-packet arrival time (mSec)
33	*tcprtt*	F	The sum of 'synack' and 'ackdat' of the TCP.
34	*synack*	F	The time between the SYN and the SYN_ACK packets of the TCP.
35	*ackdat*	F	The time between the SYN_ACK and the ACK packets of the TCP.

F. The additional features from the matched features

The generation details of the twelve additional features of the UNSW-NB15 data set (e.g., Table VI) from the matched features (e.g., Tables II-IV) are provided. Table VI is divided into two parts according to the nature and purpose of the additional generated features. The features from 36-40, are considered as general purpose features whereas from 41-47, are labelled as connection features. In the general purpose features, each feature has its own purpose, according to the defence point of view, whereas connection features are solely created to provide defence during attempt to connection scenarios. The attackers might scan hosts in a capricious way. For example, once per minute or one scan per hour [12]. In order to identify these attackers, the features 36-47 of Table VI are intended to sort accordingly with the last time feature to capture similar characteristics of the connection records for each 100 connections sequentially ordered.

TABLE VI. ADDITIONAL GENERATED FEATURES

#	Name	T	Description
General purpose features			
36	*is_sm_ips_ports*	B	If source (1) equals to destination (3)IP addresses and port numbers (2)(4) are equal, this variable takes value 1 else 0

37	*ct_state_ttl*	I	No. for each state (6) according to specific range of values for source/destination time to live (10) (11).
38	*ct_flw_http_mthd*	I	No. of flows that has methods such as Get and Post in http service.
39	*is_ftp_login*	B	If the ftp session is accessed by user and password then 1 else 0.
40	*ct_ftp_cmd*	I	No of flows that has a command in ftp session.
Connection features			
41	*ct_srv_src*	I	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
42	*ct_srv_dst*	I	No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26).
43	*ct_dst_ltm*	I	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
44	*ct_src_ ltm*	I	No. of connections of the same source address (1) in 100 connections according to the last time (26).
45	*ct_src_dport_ltm*	I	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
46	*ct_dst_sport_ltm*	I	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
47	*ct_dst_src_ltm*	I	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).

G. *The labelled features*

To label this data set, the IXIA tool has generated report about the attack data. This report is configured in the shape of the ground truth table to match all transaction records. This table consists of eleven attributes, e.g. (start time, last time, attack category, attack subcategory, protocol, source address, source port, destination address, destination port, attack name and attack reference). This data set is labelled as listed in Table VII, attack categories (i.e., *attack_cat*) and *label* for each record either 0 if the record is normal and 1 if the record is attack.

TABLE VII. LABELLED FEATURES

#	Name	T	Description
48	*attack_cat*	N	The name of each attack category. In this data set, nine categories (e.g., Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms)
49	*Label*	B	0 for normal and 1 for attack records
***Type (T.) N:** nominal, **I:** integer, **F:** float, **T:** timestamp* and ***B:** binary*			

V. DATA SET RECORDS DISTRIBUTION

Table VIII represents the distribution of all records of the UNSW-NB15 data set. The major categories of the records are normal and attack. The attack records are further classified into nine families according to the nature of the attacks.

TABLE VIII. DATA SET RECORD DISTRIBUTION

Type	No. Records	Description
Normal	2,218,761	Natural transaction data.
Fuzzers	24,246	Attempting to cause a program or n-etwork suspended by feeding it the randomly generated data.
Analysis	2,677	It contains different attacks of port scan, spam and html files penetrations.
Backdoors	2,329	A technique in which a system security mechanism is bypassed stealthily to access a computer or its data.
DoS	16,353	A malicious attempt to make a server or a network resource unavailable to users, usually by temporarily interrupting or suspending the services of a host connected to the Internet.
Exploits	44,525	The attacker knows of a security problem within an operating system or a piece of software and leverages that knowledge by exploiting the vulnerability.
Generic	215,481	A technique works against all block-ciphers (with a given block and key size), without consideration about the structure of the block-cipher.
Reconnaissa-nce	13,987	Contains all Strikes that can simulate attacks that gather information.
Shellcode	1,511	A small piece of code used as the payload in the exploitation of software vulnerability.
Worms	174	Attacker replicates itself in order to spread to other computers. Often, it uses a computer network to spread itself, relying on security failures on the target computer to access it.

VI. COMPARISON OF THE KDDCUP99 AND UNSW-NB15 DATA SET

Table IX shows a comparative analysis among the KDDCUP99 and UNSW-NB15 data sets. The table consists of eight parameters are the number of networks, number of unique ip address, type of data generation, duration of the data generation and its output format, attack vectors and the tools that are used to extract the features and the number of features for each data set. It can be observed that UNSW-NB15 data set has different attack families which ultimately reflect modern low foot print attacks.

TABLE IX. COMPARISON OF KDD CUP 99 AND UNSW-NB15

#	Parameters	KDDCUP99 [7]	UNSW-NB15
1	No. of networks	2	3
2	No. of distinct ip address	11	45
3	Simulation	Yes	Yes
4	The duration of data collected	5 weeks	16 hours 15 hours
5	Format of data collected	3 types (tcpdump, BSM and dump files)	Pcap files
6	Attack families	4	9
7	Feature Extraction tools	Bro-IDS tool	Argus, Bro-IDS and new tools.
8	No. of features extraction	42	49

VII. Final Shape of the UNSW-NB15 Data Set Files

In this section, the description of the final shape of the UNSW-NB15 is provided. The purpose of this section is to guide the researchers on how to use and manipulate final CSV files of the UNSW-NB15 data set. Four CSV files of the data records are provided and each CSV file contains attack and normal records. The names of the CSV files are *UNSW-NB15_1.csv*, *UNSW-NB15_2.csv*, *UNSW-NB15_3.csv* and *UNSW-NB15_4.csv*.

In each CSV file, all the records are ordered according the last time attribute. Further, the first three CSV files each file contains *700000* records and the fourth file contains *440044* records. The ground truth table is named *UNSW-NB15_GT.csv*. The list of event file is labelled *UNSW-NB15_LIST_EVENTS* which contains attack category and subcategory. The interested reader can obtain the raw pcap files by e-mailing the authors.

VIII. Conclusion and Future Work

In this paper, the existing benchmark datasets are not representing the comprehensive representation of the modern orientation of network traffic and attack scenarios. UNSW-NB15 is created by establishing the synthetic environment at the UNSW cyber security lab. The key utilised IXIA tool, has provided the capability to generate a modern representative of the real modern normal and the synthetical abnormal network traffic in the synthetic environment. UNSW-NB15 represents nine major families of attacks by utilising the IXIA PerfectStorm tool. There are 49 features that have been developed using Argus, Bro-IDS tools and twelve algorithms which cover characteristics of network packets. In contrast the existing benchmark data sets such as KDD98, KDDCUP99 and NSLKDD, realised a limited number of attacks and information of packets which are outdated. Moreover, the UNSW-NB15 is compared with KDDCUP99 data set by considering some key features and it shows the benefits. In future, it is expected that, the UNSW-NB15 data set can be helpful to the NIDS research community and considered as a modern NIDS benchmark data set.

Acknowledgment

This work is supported by cyber range lab of the Australian Centre for Cyber Security (ACCS) at UNSW in Canberra. The authors are grateful for the manager of the Cyber range lab.

References

[1] R.Heady, G.Luger, A.Maccabe, M.Servilla. “The architecture of a network level intrusion detection system”. Tech. rep., Computer Science Department, University of New Mexico, New Mexico ,1990.

[2] M.Aydın, M. Ali, A. Halim Zaim, and K. Gökhan Ceylan. “A hybrid intrusion detection system design for computer network security”, Computers & Electrical Engineering, 2009, p 517-526.

[3] Axelsson, Stefan. “Intrusion detection systems: A survey and taxonomy”, Technical report, 2000, Vol. 99.

[4] C.Dartigue, H.Jang and W.Zeng, ”A new data-mining based approach for network intrusion detection”, Communication Networks and Services Research Conference. CNSR'09. Seventh Annual. IEEE, 2009, p 372-377.

[5] J.Zhang, and Z.Mohammad, “Anomaly based network intrusion detection with unsupervised outlier detection”, Communications, 2006. ICC'06. IEEE International Conference on. Vol. 5. IEEE.

[6] P.Gogoi et al, ”Packet and flow based network intrusion dataset."Contemporary Computing”. Springer Berlin Heidelberg, 2012. P 322-334.

[7] KDDCup1999.Available-on: http://kdd.ics.uci.edu/databases/kddcup99/KDDCUP99.html, 2007.

[8] NSLKDD. Available on: http://nsl.cs.unb.ca/NSLKDD/, 2009.

[9] McHugh, John, ”Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory”. ACM transactions on Information and system Security, 3, 2000, p 262-294.

[10] V.Mahoney, and K.Philip, “An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection."Recent Advances in Intrusion Detection”. Springer Berlin Heidelberg, 2003.

[11] A.Vasudevan, E. Harshini, and S. Selvakumar, "SSENet-2011: a network intrusion detection system dataset and its comparison with KDD CUP 99 dataset”, Internet (AH-ICI), 2011, Second Asian Himalayas International Conference on. IEEE.

[12] M.Tavallaee, E.Bagheri, W.Lu, and A.Ghorbani, “A detailed analysis of the KDD CUP 99 data set”. Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications, 2009.

[13] DARPA98.Available on:http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/, 1998.