Big Data Analytics

(Resit Assessment)

Submitted by:
Joanna Marie Diaz
joannamarie838@gmail.com

Table of Contents

Big Data Analysis with Hadoop and Spark (UNSW_NB15 dataset)

Abstract – *This assessment provides a brief introduction to Big Data and Hadoop frameworks that is used for Big Data Analytics. Furthermore some description of the query language (Hive) and computing framework (Spark) in the Hadoop ecosystem. The analytical tools were used to explore the UNSW_NB15 dataset and identify relevant information that could assist in training a model that can detect a network intrusion. Cloudera interface was used to interact with Hadoop ecosystem and perform exploratory analysis on the said dataset using Hue and HiveQL. On the contrary, Google's Colaboratory was utilize to perform data analysis in Spark and it's Python API, PySpark to conduct data analysis and train a Binary Classification and Multi-class Classification model.*

I. Introduction

Data can be a concept of information and event that is being performed on daily routines. For years, data management has been a challenge for industrialists and many innovators ponder a method of how to better store and analyze data. When the huge amount of data had begun to explode, people in technical and business industries began to cultivate and refine previous technologies that gradually surmount the challenges of how to make the processing of data effortless and more efficient. From punch cards and magnetic storage to cloud data storage, many industrialists have taken advantage of this technology to develop software and build frameworks that can conserve and utilize the information that is occurring every second.

This technology gives way to the creation of analytical tools that can easily yield descriptive, predictive, and prescriptive analysis of the data. consequently, the huge amount of data is now seen as an investment for discerning a hidden value that can emerge by exerting these analytical tools.

II. Big Data

According to the Oxford dictionary, data are defined as:

"the quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media".

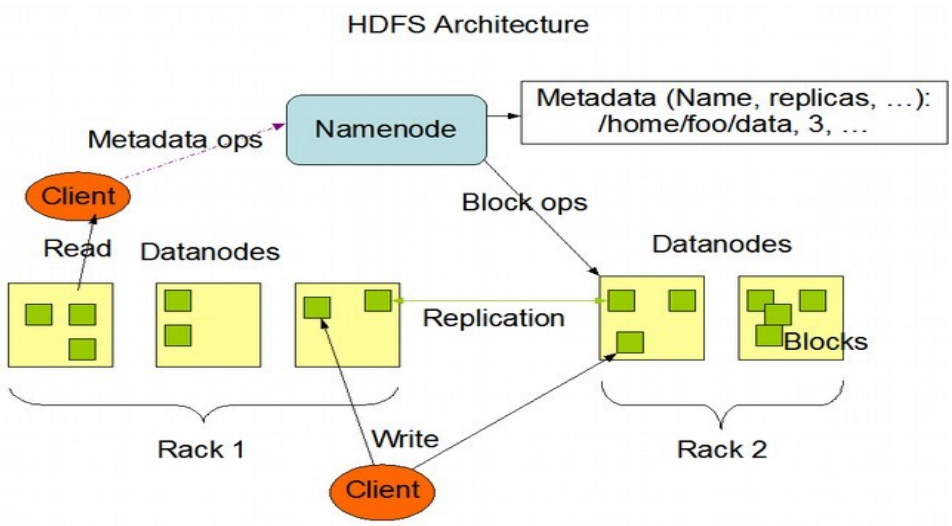
This data was a collection of information that must be preserved as a record and basis for future research. Until newer technology has begun to emerge and data become uncontrollable. Around 2005, according to Chambers and Zaharia (2018), the production of individual processors stopped due to heat dissipation. At that moment instead of recreating a new version of more faster CPU, the hardware developer switched toward adding parallel CPUs that run at the same speed. Due to this, applications that run on these computing devices also need to be altered to adapt to the processor's speed and current architecture. Furthermore many trending technologies that generate data has also improved and became more affordable, thus more and more data has been accumulated. In the same year, Rouger Mougalas called this huge amount of data as Big Data.

Big data has been characterized by its three key properties which are, volume, velocity, and variety. Sedkaoui (2018) elucidate the 3V's as, volume, the size of the data that is being gathered and processed. Velocity, speed, and rate of the growth of data. Variety: refers to heterogeneity, representation, and interpretation of the unveiled data. He also quotes, "An information asset whose volume is large, velocity is high, and formats are various."

Data nowadays come from different resources. From medical health records, and bank transactions to airport flight details. But a larger amount of generated data every day emerge through the internet, like social networks, IoT, emails, mobile apps, etc. In 2005 Hadoop was also released to tackle the challenges of big data management.

III. Hadoop

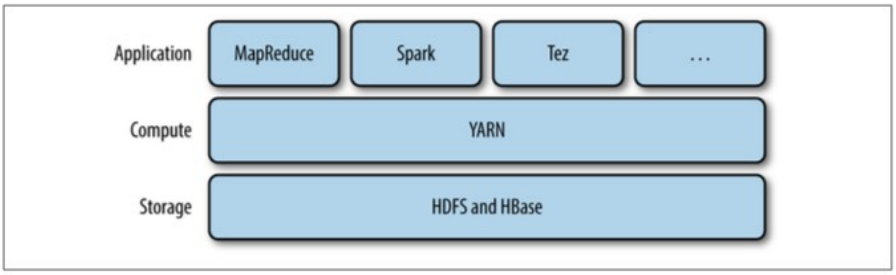
Before the explosion of big data, the notion of data storage was a relational database that rely on structured data processing that is being explored on a day-to-day basis by a diverse corporation that hinges on daily records of each task and individual that is related to the firm. Even for such amounts of data, data exploitation and manipulation appear as a challenge by using traditional databases and structured processing software. The need for databases to store, manage and manipulate structured and unstructured data is conclusive to tackle in this new era. Traditional storage is not enough to accommodate the huge amount of data came from various resources of technology.
Hadoop was created based on an open-source software framework called Nutch, and was merged into google's MapReduce. Hadoop and the frameworks that have been developed to support Hadoop's functionality is an open source framework that can process structured and unstructured data, from nearly all digital sources. Hadoop is mainly designed for data batch processing.
The Hadoop ecosystem arises as a cost-effective solution dealing with large data sets. It imposes a particular programming model, called MapReduce, that fragments computation tasks into units allowing these blocks to be distributed around a cluster of commodity hardware, thereby providing cost-effective, horizontal scalability. Underneath this computation, the model is a distributed file system called the Hadoop Distributed Filesystem (HDFS). (Capriolo et al, 2012). MapReduce is a batch query processor, it can run an ad hoc query in the whole datasets in a reasonable time (White 2012).
The components of Hadoop were built to tackle the handling of Big Data. One is to provide storage with a concept of distributed data on numerous computers. The architecture of Hadoop's HDFS (Hadoop Distributed File System) is to divide data into smaller blocks and distribute them to different clusters while allowing the data to be processed and analyzed at the same time. One of the main features of HDFS is its replication method, which makes it fault-tolerant. Allowing the data to be replicated and retained in another cluster, Hence even if one DataNode crashes, data is not lost regardless.



(Source: https://hadoop.apache.org/)

Before the storage of data, these data are processed by MapReduce which provides a computing programming paradigm that processes data in parallel. Distributing each block of data on the different clusters while providing mapping computation on each split data on DataNodes. The data will then be aggregated to produce the final output. Thus providing a more efficient time for data processing because of its parallelism computation.

The third component of Hadoop handles the scheduling and computation of resources for the distributed and processed data which is called YARN (Yet Another Resource Negotiator). It also supports other distributed paradigms aside from MapReduce. YARN process job request and manages cluster resources on the Hadoop ecosystem.

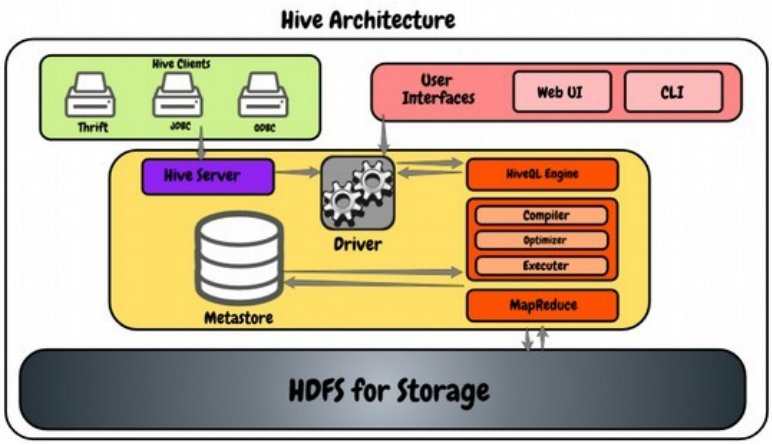


(Yarn Application)
Hadoop: The definitive guide (White 2012)

The Hadoop ecosystem comprises tools and frameworks that manage Big Data processing, which includes data querying programming tools like Pig, Hive, Spark, etc. For this assessment, two of these data querying and programming tools and framework were used.

IV. Apache Hive

Hive is a framework that is created on top of Hadoop to provide ease of big data queries. It is a SQL-like language that supports structured data query that is stored on a table. Hence the adaptation of HiveQL as a query language for Hadoop has been supported by a large base of SQL users. Hive translates most queries to MapReduce jobs, thereby exploiting the scalability of Hadoop while presenting a familiar SQL abstraction (Capriolo et al, 2012).

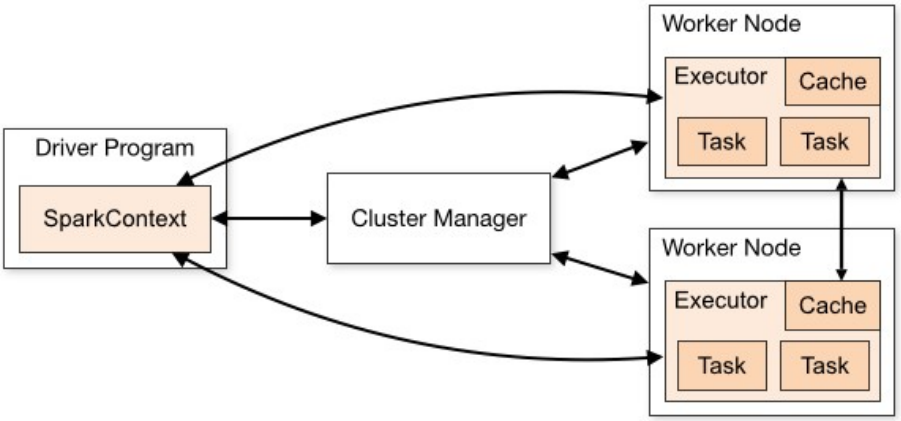


(Source: https://www.analyticsvidhya.com/)

V. Apache Spark

Apache Spark is a unified computing engine for large amounts of data that compose of sets of libraries for parallel processing on a computer cluster. It supports the most popular programming languages (Python, Java, Scala, and R). Spark was designed to support an extensive range of data analytical tasks. From loading of data, to SQL queries to machine learning and streaming computation. (Chambers and Zaharia, 2018).

The creation of Spark has started at UC Berkeley in 2009 as a research project to tackle multiple parallel programming engines for clusters. At that time Hadoop's MapReduce was the dominant parallel computing paradigm that process data on thousand of clusters. The Spark team began to study the MapReduce functionality and created various development based on the engine. Spark initially was built for batched data processing but eventually become a resource for multiple data processing and analysis. It implements the adaptation of several libraries that makes Spark a powerful framework for Big Data management.
Despite the fact that Spark is a part of Hadoop's ecosystem, it can be a standalone platform by leveraging other resource managers outside Hadoop. Spark also claims that it can run 100 times faster than MapReduce.
What makes Spark different from Hadoop's MapReduce is that Spark utilizes DAG (Direct Acyclic Graph) that provides in-memory computation that schedule and execute a task in the memory. It is also a fault-tolerant through RDDs which are designed to handle the failure of any worker node in the cluster.



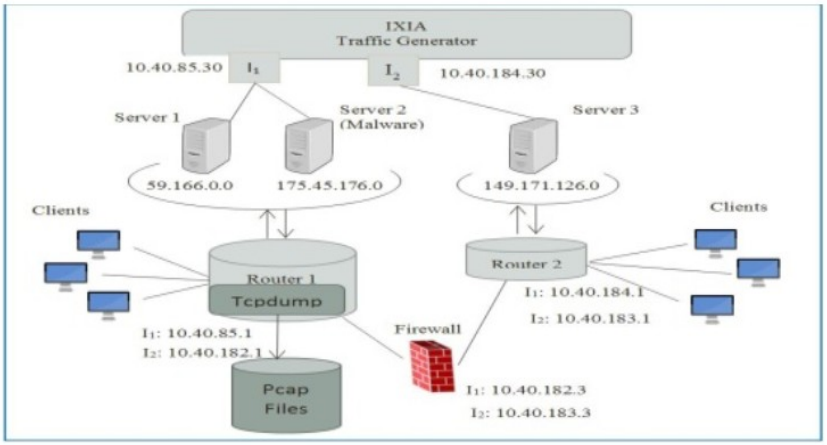
(Source: https://spark.apache.org/)

Spark has several core abstractions: Datasets, DataFrames, SQL Tables, and Resilient Distributed Datasets (RDDs). These different abstractions all represent distributed collections of data. The easiest and most efficient are DataFrames, which are available in all languages (Chambers and Zaharia, 2018).

VI The Dataset

UNSW-NB15 was developed in 2015 (Moustafa and Slay, 2015) to provide a more comprehensive dataset for Network Intrusion Detection System (NIDS) which can mirror modern network traffic scenarios, extensive variation of low footprint intrusions, and the depth structure of information about the network traffic.

The dataset is composed of 2,540,044 realistic modern normal and abnormal network activities. These records were accumulated by the IXIA Perfect-storm at cyber range Lab at the Austrian Centre for Cyber Security (ACCS).
The traffic generator uses three virtual servers. The two servers were arranged to produce normal traffic to the host and the third one is configured to generate abnormal network traffic (Zoghi and Serpen, 2021).



(The Testbed Visualization for UNSW-NB15, Moustafa and Slay, 2015)

VI.1 UNSW-NB15 Dataset

For this activity, UNSW-NB15 datasets were utilized as an experiment for big data analysis and visualization using Hive and Spark queries.
UNSW-NB15 datasets were composed of 49 network features. The dataset was extracted through a tcdump tool to produce a pcap file. The dataset was produced to test a Network Intrusion Detection System (NIDS) application that monitors network traffic for malevolent purposes. Several studies were conducted to create a robust system for the Network intrusion detection system using the said dataset. The UNSW-NB15 dataset is considered one of the current instruments to test the effectiveness of the Network Intrusion Detection System to identify any possible network attacks.

VII HiveQL Data Analysis

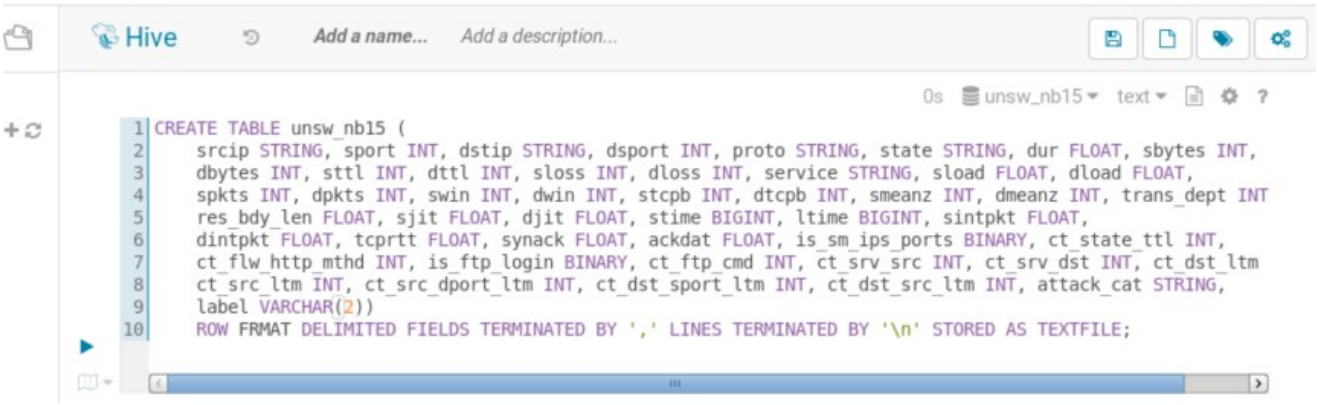
HiveQL was used to perform exploratory analysis on known columns and rows of the UNSW_NB15 dataset. It includes the exploration of Hive's built-in functions: Date, Math, Conditional, and String methods.

VII.1 Working on the Data Using Apache Hive

VII.1.1 Creating Schema

```
1 CREATE SCHEMA UNSW_NB15;
```

VII.1.2 Create table

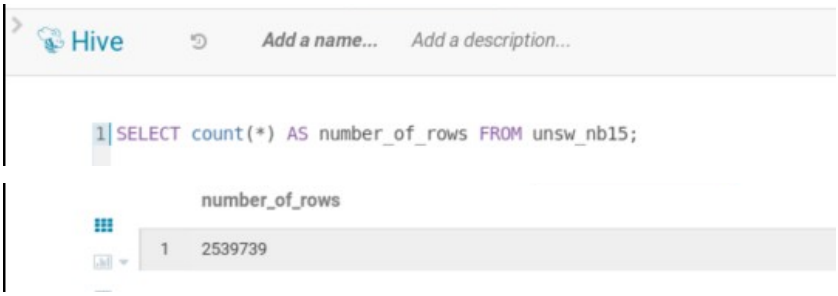


VII.1.3 Load data to HDFS

```
LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/unsw_nb15_datasets/UNSW-NB15.csv'
OVERWRITE INTO TABLE unsw_nb15;
```

VII.1.4 Hive Simple Query

Number of rows



- Based on the query, the dataset composed of 2,539,739 rows with 49 columns.

Limit clause

```
SELECT srcip, dstip, proto, state, dur, sbytes, dbytes, sttl, dttl, service, spkts, dpkts
stcpb, dtcpb, sjit, djit, stime, ltime, tcprtt, attack_cat, label
FROM unsw_nb15 LIMIT 20;
```

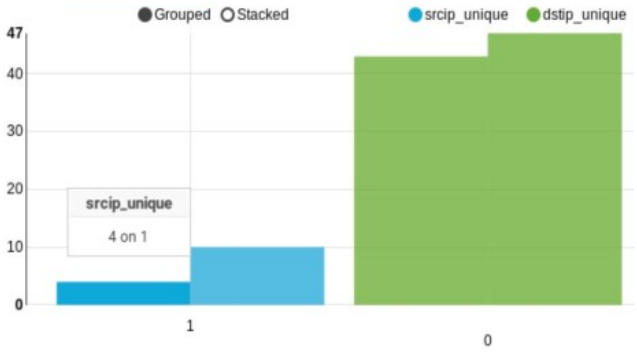
	srcip	dstip	proto	state	dur	sbytes	dbytes	sttl	dttl	service	spkts	stcpb
1	59.166.0.3	149.171.126.8	tcp	FIN	0.82546001672744751	7812	16236	31	29	-	122	126
2	59.166.0.0	149.171.126.6	tcp	FIN	0.10181500017642975	4238	65628	31	29	-	72	74
3	59.166.0.5	149.171.126.2	tcp	FIN	0.044002998620271683	2750	29104	31	29	-	44	48
4	59.166.0.9	149.171.126.0	tcp	FIN	2.7908298969268799	10476	395734	31	29	-	180	320
5	59.166.0.8	149.171.126.9	tcp	FIN	2.6335000991821289	13350	548216	31	29	-	232	438
6	59.166.0.3	149.171.126.3	tcp	FIN	0.11504799872636795	1958	2308	31	29	-	22	24
7	59.166.0.7	149.171.126.0	udp	CON	0.0033619999885559082	146	178	31	29	dns	2	2
8	59.166.0.3	149.171.126.9	tcp	FIN	0.45305201411247253	424	8824	31	29	ftp-data	8	12

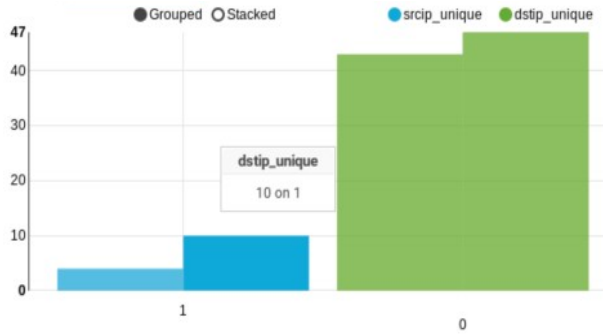
Groupby by label

```sql
SELECT label,
    count(DISTINCT srcip) AS srcip_unique,
    count(DISTINCT dstip) AS dstip_unique
FROM unsw_nb15
GROUP BY label;
```

Query History | Saved Queries | Results (2)

	label	srcip_unique	dstip_unique
1	1	4	10
2	0	43	47





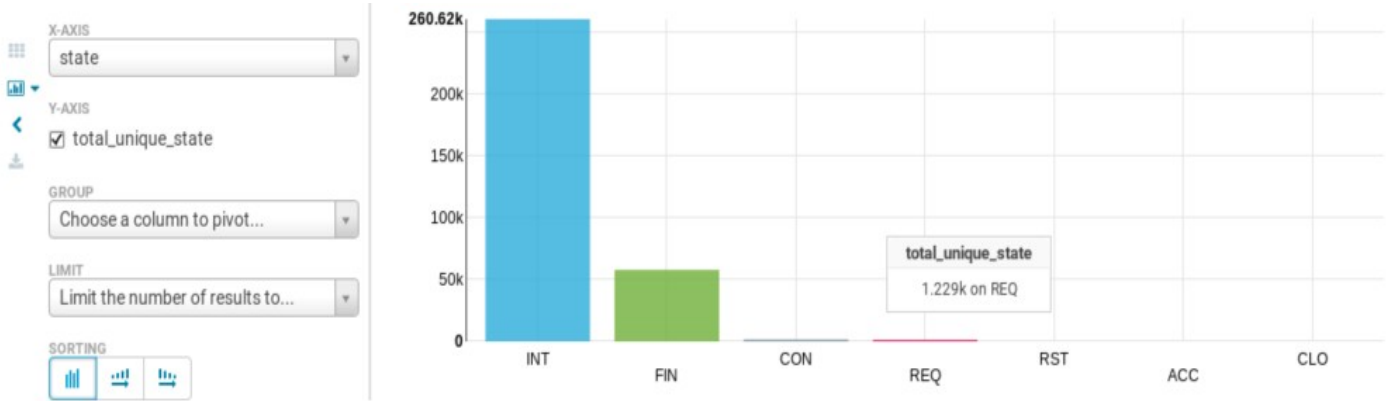
- This shows that on 47 source IP address, 43 were counted as normal and 4 were labeled as abnormal or attack. While on destination IP address, 47 were normal and 10 were counted as abnormal or attack.

Groupby and orderby on state column

```sql
SELECT state, count(state) AS total_unique_state
FROM unsw_nb15
WHERE label=1
GROUP BY state
ORDER BY total_unique_state DESC;
```

Query History | Saved Queries | Results (7)

	state	total_unique_state
1	INT	260620
2	FIN	57923
3	CON	1495
4	REQ	1229
5	RST	13
6	ACC	2
7	CLO	1



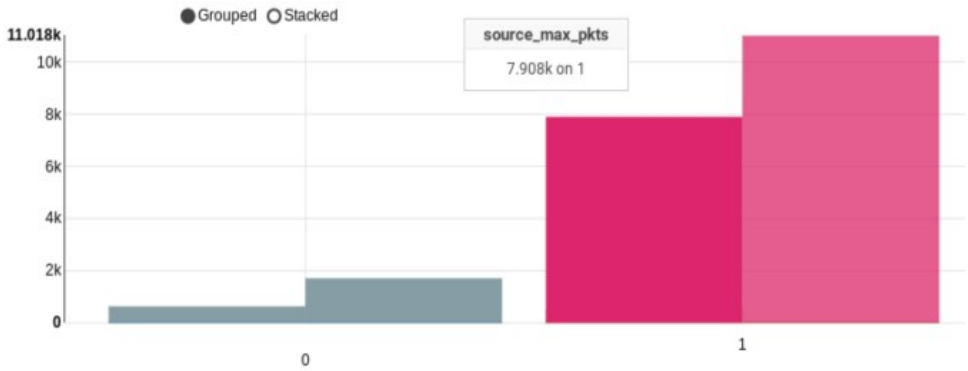
- The query shows that there are 7 states that where use to simulate network abnormalities, where the four evident services are INT variable with 260,620 connection, FIN: 57,923, CON: 1,495, and REQ: 1,229.

Where Condition

```
SELECT
    proto, state,
    max(spkts) AS source_max_pkts,
    max(dpkts) AS dest_max_pkts,
    min(spkts) AS source_min_pkts,
    min(dpkts) AS dest_min_pkts,
    label
FROM unsw_nb15
WHERE state BETWEEN 'FIN' AND 'INT' AND service<>'-'
GROUP BY proto, state, label
ORDER BY proto DESC, state, label;
```

Query History | Saved Queries | Results (4)

	proto	state	source_max_pkts	dest_max_pkts	source_min_pkts	dest_min_pkts	label
1	udp	INT	16	0	1	0	0
2	udp	INT	512	0	1	0	1
3	tcp	FIN	642	1716	1	1	0
4	tcp	FIN	7908	11018	0	2	1



- The query shows that the transferred and received packets for the abnormal transaction (attack transaction) with udp and tcp as protocol has more than 10 times the packets value compared to the normal network transaction.

VII.1.5 Hive Built-in Function

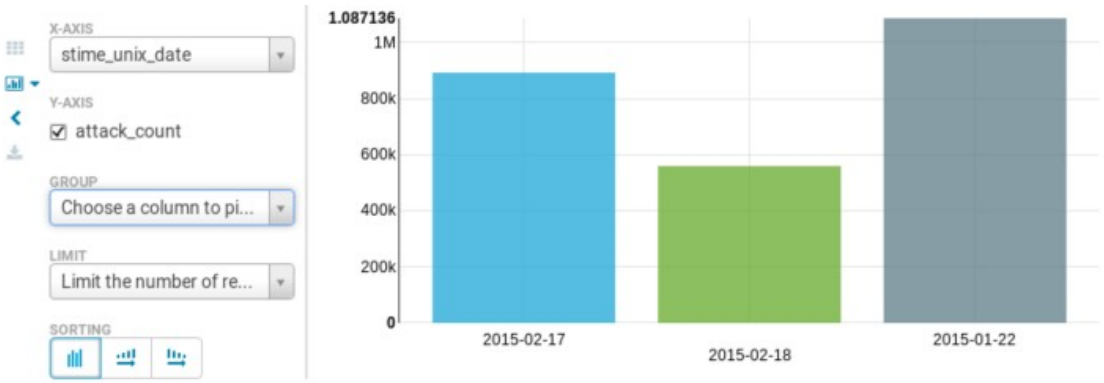
A. Date Function

> From_unixtime()

```
1 SELECT from_unixtime(stime, 'yyyy-MM-dd') AS stime_unix_date, count(attack_cat) AS attack_count
2 FROM unsw_nb15
3 GROUP BY from_unixtime(stime, 'yyyy-MM-dd');
```

1m, 55s unsw_nb15 text

	stime_unix_date	attack_count
1	2015-02-17	892739
2	2015-02-18	559864
3	2015-01-22	1087136



- This shows the number of attacks that were simulated during the specific date of observation which consist of three days. Having 892,739 on February of 17th of 2015. 559,864 on the 18th and 1,087,136 on January 22 of 2015.

According to Moustafa and Slay 2015:

"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".

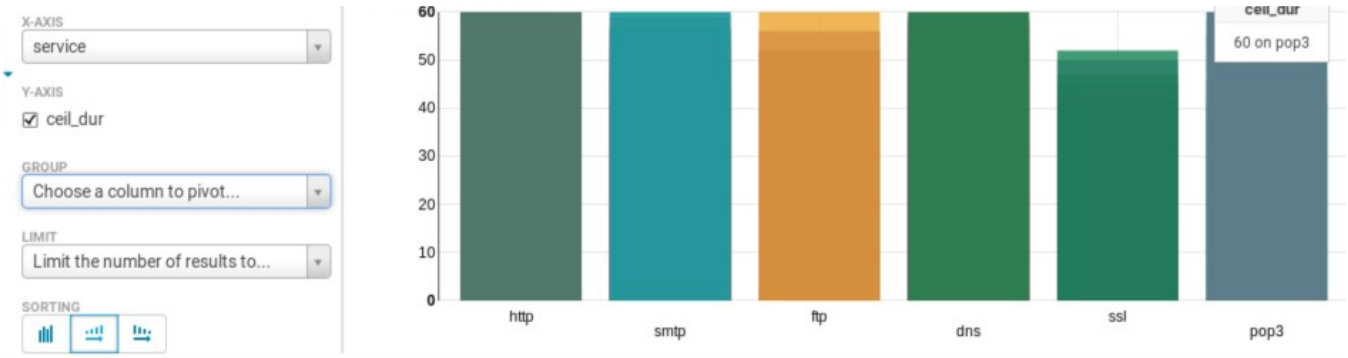
B. Mathematical Function

> ceil()

```
1 SELECT attack_cat, service, ceil(dur) AS ceil_dur
2 FROM unsw_nb15
3 WHERE attack_cat<>'' AND service<>'-'
4 GROUP BY attack_cat, service,  ceil(dur)
5 HAVING ceil_dur > 5
6 ORDER BY ceil_dur DESC, attack_cat, service;
```

Query History Saved Queries Results (319)

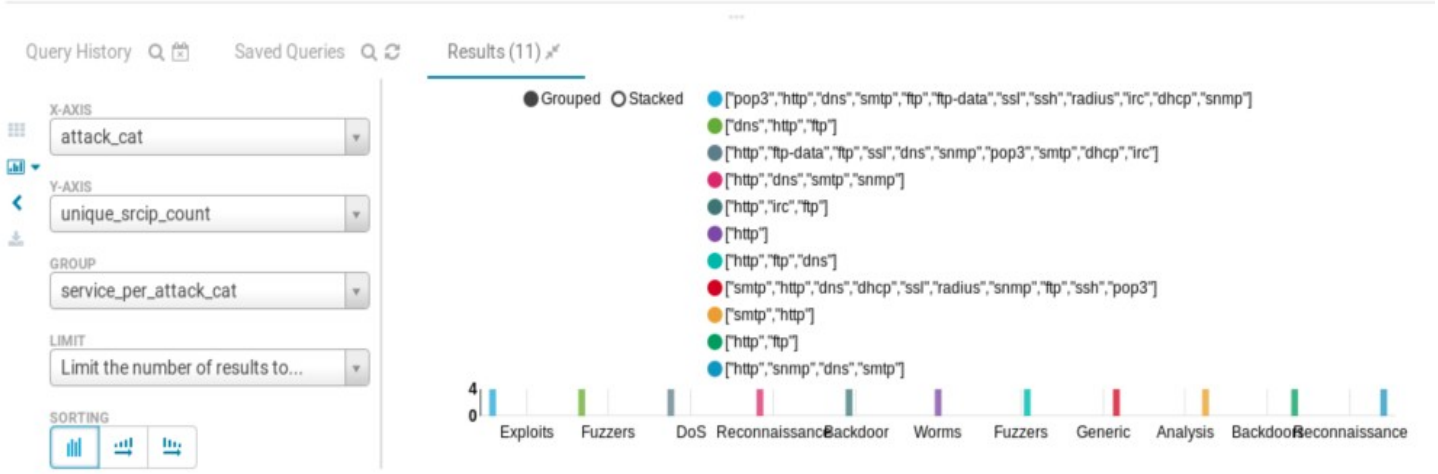
	attack_cat	service	ceil_dur
1	Analysis	http	60
2	DoS	http	60
3	DoS	pop3	60
4	DoS	smtp	60
5	Exploits	dns	60
6	Exploits	http	60
7	Exploits	smtp	60
8	Fuzzers	dns	60
9	Fuzzers	ftp	60

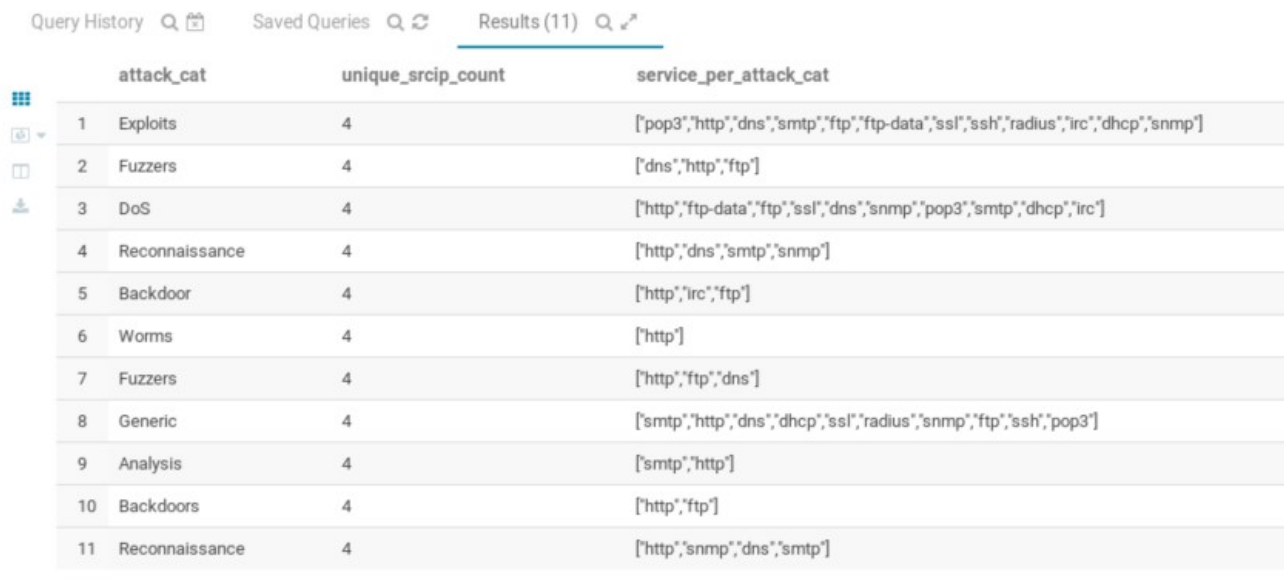


- The result were composed of 319 results that list the attack category versus the services that has been used with the maximum duration of 60 second and minimum duration more than 5 seconds.

> collect_set()

```
SELECT attack_cat,
    count(DISTINCT srcip) AS unique_srcip_count,
    collect_set(service) AS service_per_attack_cat
FROM unsw_nb15
WHERE label = '1' AND service <> '-'
GROUP BY attack_cat;
```





	attack_cat	unique_srcip_count	service_per_attack_cat
1	Exploits	4	["pop3","http","dns","smtp","ftp","ftp-data","ssl","ssh","radius","irc","dhcp","snmp"]
2	Fuzzers	4	["dns","http","ftp"]
3	DoS	4	["http","ftp-data","ftp","ssl","dns","snmp","pop3","smtp","dhcp","irc"]
4	Reconnaissance	4	["http","dns","smtp","snmp"]
5	Backdoor	4	["http","irc","ftp"]
6	Worms	4	["http"]
7	Fuzzers	4	["http","ftp","dns"]
8	Generic	4	["smtp","http","dns","dhcp","ssl","radius","snmp","ftp","ssh","pop3"]
9	Analysis	4	["smtp","http"]
10	Backdoors	4	["http","ftp"]
11	Reconnaissance	4	["http","snmp","dns","smtp"]

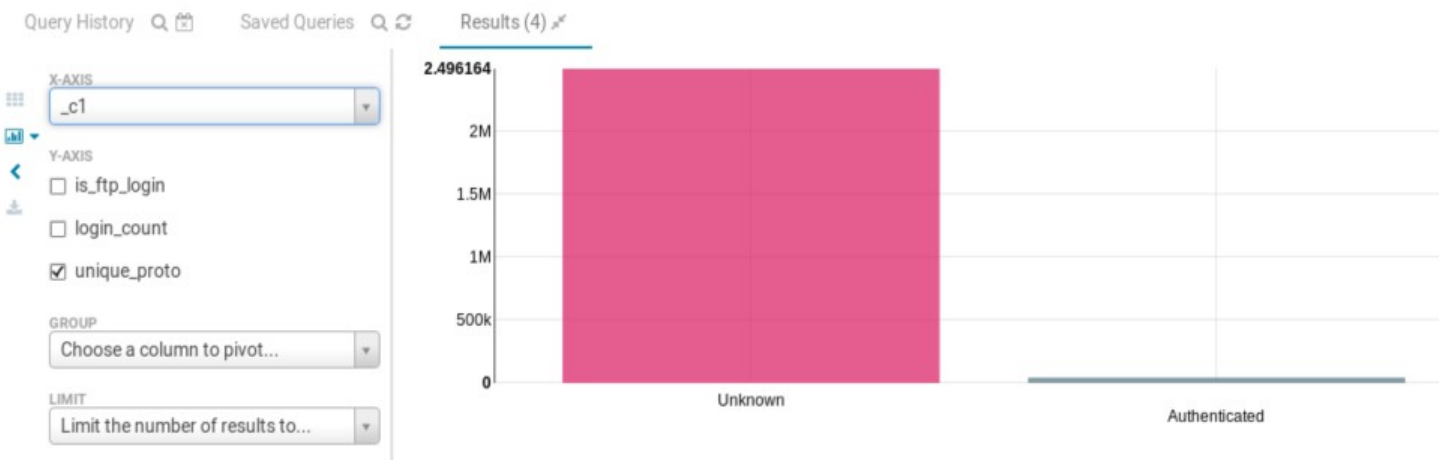
- Uses collect_set to get the services that has been used on every attack instances on the four unique source IP address.
The data shows that on each attack category, it has a prevalent service of “http” as a protocol for network connection.

C. Conditional Function
> Case ()

```
SELECT
    is_ftp_login,
        CASE is_ftp_login WHEN 1 THEN 'Authenticated' ELSE 'Unknown' END,
    count(DISTINCT is_ftp_login) AS login_count,
    count(proto) AS unique_proto
FROM unsw_nb15
GROUP BY is_ftp_login
ORDER BY unique_proto;
```

Query History | Saved Queries | Results (4)

	is_ftp_login	_c1	login_count	unique_proto
1	2	Unknown	1	30
2	4	Unknown	1	156
3	1	Authenticated	1	43389
4	0	Unknown	1	2496164



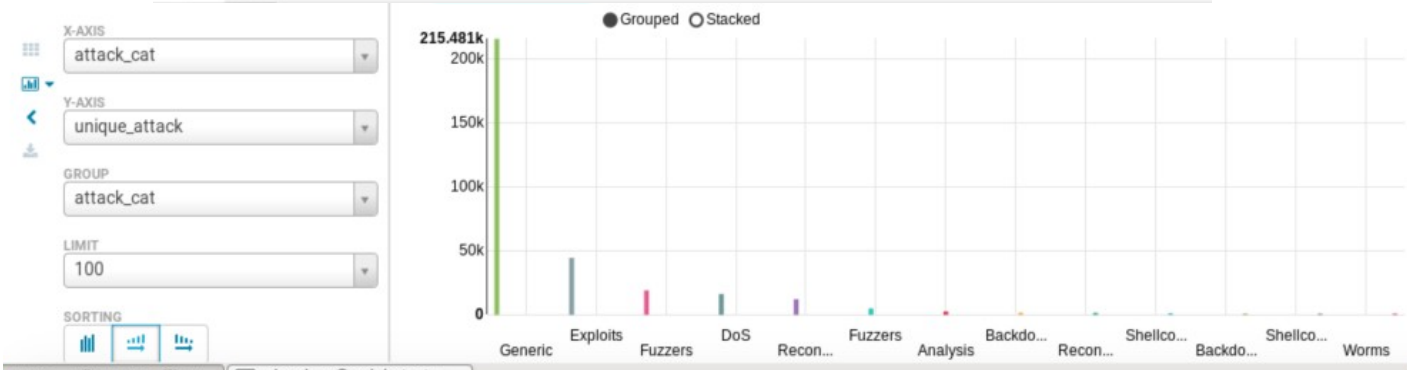
- The query shows that there are 2,496,350 ftp unknown login and 43,389 authenticated ftp login.

> if()

```
SELECT
    if(attack_cat='','Normal',attack_cat) AS attack_cat,
    count(attack_cat) AS unique_attack
FROM unsw_nb15
GROUP BY if(attack_cat='','Normal',attack_cat)
ORDER BY unique_attack DESC, attack_cat;
```

Query History | Saved Queries | Results (14)

	attack_cat	unique_attack
1	Normal	2218456
2	Generic	215481
3	Exploits	44525
4	Fuzzers	19195
5	DoS	16353
6	Reconnaissance	12228
7	Fuzzers	5051



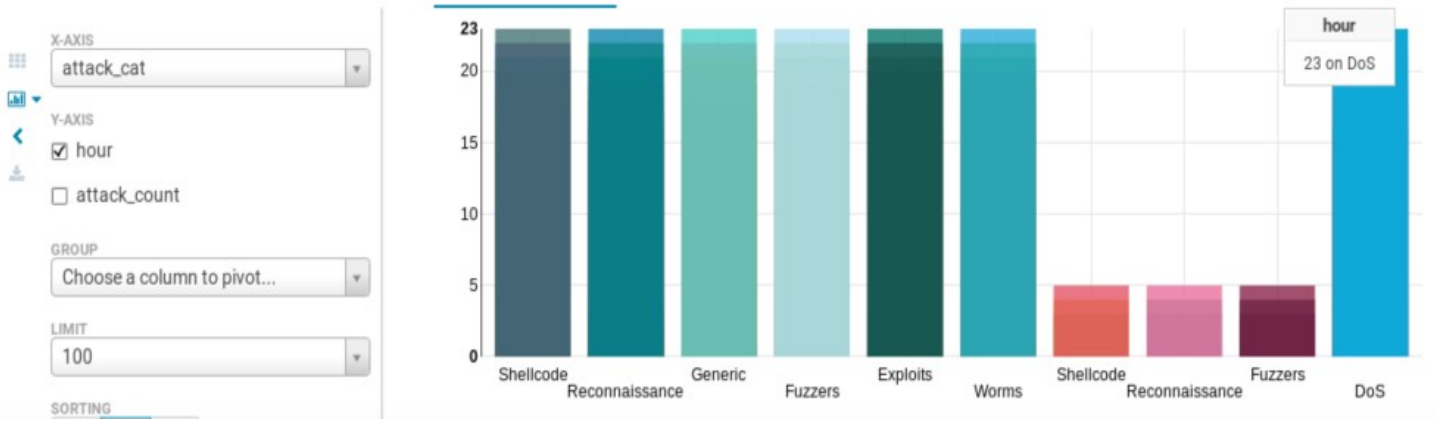
- The result suggest a great variance between the normal and attack transaction representing the movement of an authentic network activity.

D. String Function

>substr()

```
SELECT attack_cat,
    hour(substr(from_unixtime(stime, 'MM-dd-yyyy HH:mm:ss'),12)) AS hour,
    count(DISTINCT attack_cat) AS attack_count
FROM unsw_nb15
WHERE label='1'
GROUP BY attack_cat, hour(substr(from_unixtime(stime, 'MM-dd-yyyy HH:mm:ss'),12))
ORDER BY attack_cat DESC, hour DESC;
```

Query History | Saved Queries | Results (133)



- The query shows that the simulation that were conducted for attack category mostly occurred at 11pm.

Despite that there are some peculiarity on the dataset that needs further speculation, there are extracted information that is informative and evident,

- The date and hours of the dataset show the span of creation of the dataset, and prove the specific date of accumulation.
- The classification and number of different attack categories versus the services and protocol that have been used during the attack simulation.
- The great difference in amount between normal and abnormal connections.

VIII PySpark Data Analysis

PySpark is an interface that utilizes Python programming language in Apache Spark. It implements a Python API to create Spark applications and supports most of Spark's features such as Spark SQL, Dataframe, Streaming, Mllib (Machine Learning), and Spark Core.
For this part of the assessment, Google's Colaboratory was used to set up the Spark environment where data analysis and ML training was performed. The provided UNSW_NB15 dataset is composed of 49 columns and 2,539,738 rows.

VIII.1 Preprocessing

To avoid mis-leading result, dataset have to undergone data cleaning or preprocessing. For this part of the assessment the following step are applied to ensure the quality of the data and to produce the final training set.

- Dropping of duplicate rows
- Filling missing values
- Selecting relevant features
- One hot encoding / Normalization
- Feature extraction

VIII.2 Correlation Analysis

On this step, the selected and extracted features undergone the correlation method to find the linear relationship between the feature and label column. I set the value of selecting the correlated value between two variable from more than zero and less than one. I hereby after use the column with positive relationship with the label as the selected features for training the model.

VIII.3 Training the Model

VIII.3.1 *Multi-class Classification Model*

For this step, two machine learning classification tasks were performed on the dataset. I set the multi-class classification task to identify the missing value on the service column. For this task, I use the Decision Tree and Random Forrest to determine what values fit on those missing values.

Classifier	Evaluation Result	With Cross Validation
Decision Tree	97.24%	99.98%
Random Forest	99.45%	99.71%

As the result show, the Decision Tree Classifier with 5-fold Cross Validation has much higher accuracy. Accordingly, this model has been used to fill in the missing data for the un-labeled service column. A new dataset has been assembled after joining the labeled service and un-labeled service dataset without any missing value. This new dataset will now be used for the Binary Classification task.

VIII.3.2 *Binary Classification Model*

On this classification task I use the Logistic Regression and One vs Rest algorithm to train the new formed dataset. By selecting the correlated features to the label, I managed to get a good result for both classifier.

Classifier	**Accuracy Result**
Logistic Regression	Accuracy: 0.9999999800950727
One vs Rest	Accuracy: 0.9999999871304348

Logistic Regression

attack_cat	prediction	label	count
Exploits	1.0	1	8279
Fuzzers	1.0	1	5336
Shellcode	1.0	1	60
DoS	1.0	1	1723
Normal	0.0	0	587625
Analysis	1.0	1	633
Backdoor	1.0	1	527
Fuzzers	1.0	1	1221
Generic	1.0	1	7554
Worms	1.0	1	61
Reconnaissance	1.0	1	3420
Backdoors	1.0	1	78
Reconnaissance	1.0	1	492
Shellcode	1.0	1	368
Normal	1.0	0	3

One vs Rest

attack_cat	prediction	label	count
Exploits	1.0	1	8279
Fuzzers	1.0	1	5336
Shellcode	1.0	1	60
DoS	1.0	1	1723
Normal	0.0	0	587625
Analysis	1.0	1	633
Backdoor	1.0	1	527
Fuzzers	1.0	1	1221
Generic	1.0	1	7554
Worms	1.0	1	61
Reconnaissance	1.0	1	3420
Backdoors	1.0	1	78
Reconnaissance	1.0	1	492
Shellcode	1.0	1	368
Normal	1.0	0	3

Both classification has identical accuracy result. Producing an almost perfect model for the modern network intrusion detection system.

Note:
Please see github repo for complete code: https://github.com/j-anne/Big_data_analytics.git

VIII.3.3 Machine Learning Model

The following are the description for the Machine Learning Algorithm that were use for the classification method accordingly.

- **Decision Tree** – A classifier that construct a decision tree to predict a class for an observation. The aim of decision tree is to select a binary question that best splits the data into two homogenous groups at each branch of the tree, such that it minimizes the level of data entropy at the next (Theobald 2017).

- **Random Forest** – This model yield multiple decision trees (hence the name—forest) and uses the mode output of those decision trees to classify observations. The RandomForestClassifier supports both binary and multinomial labels (Drabas and Lee 2017).

- **Logistic Regression** – The benchmark model for classification. The logistic regression uses a logit function to calculate the probability of an observation belonging to a particular class. A common application of the sigmoid function is found in logistic regression.

- **One vs Rest** – A reduction of a multi-class classification to a binary one. For example, in the case of a multinomial label, the model can train multiple binary logistic regression models. For example, if label == 2, the model will build a logistic regression where it will convert the label == 2 to 1 (all remaining label values would be set to 0) and then train a binary model. All the models are then scored and the model with the highest probability wins (Drabas and Lee 2017).

IX. Alternative solution for high level languages and analytic approaches

1. Apache Flume

Apache flume is a distributed ingestion tool for collecting a large amount of streaming data from various sources in the web, aggregating them and moving them into HDFS. Streaming data can be log file, email message, sensors, etc. It is apart of Hadoop ecosystem that ingest streaming data into HDFS.

2. Apache Flink

Apache Flink is a distributed flow engine, which runs data flow (DAG) programs. This engine possesses a lot of advantages includes:

- It support both batch and stream processing.
- It is an open source framework that process large amount of data and focuses on low latency and high throughput and support horizontal scaling.
- It provides a stateful system that recognize persisting intervening processing result internally without using external resources.
- It also has built-in libraries and APIs for SQL query (Table API), Graph API (Gelly), and ML algorithm (Flink ML) that promote DataSet and DataStream API programs.
- Fault tolerant.

3. Apache Storm

Apache Storm is an open-source framework that is known for its fast-distributed real-time data processing platform. Like all platforms that implemented distributed computing paradigms, Apache Storm is also fault-tolerant and processes data in parallel while being executed. It does so by implementing a MapReduce task on its environment which is called Bolts. However, it surpasses MapReduce when it comes to speed and performance. Additionally, Storm can be adopted by any application that implements any programming language.

X. Conclusion

For this study, significance of Big Data were tackled and explored. Furthermore, the Hadoop framework and analytical tools were utilize to scrutinize the UNSW_NB15 dataset which is a dataset tool for modern network intrusion detection. Data cleaning and preprocessing on the dataset were conducted. The final dataset was used to train a Binary and Multi-class Classification model. The process ended up with an almost perfect accurate result for both Binary and Multi-class Model. This study still needs further improvement for model evaluation and data visualization.

XI. References:

Bagui, S., Kalaimannan, E., Bagui, S., Nandi, D., Pinto, A., (2019), Using machine learning techniques to identify rare cyber-attacks on the UNSW-NB15 dataset. Security and Privacy. 2019;2:e91. https://doi.org/10.1002/spy2.91

Capriolo, E., Wampler, D. and Rutherglen, J., 2012. Programming Hive: Data warehouse and query language for Hadoop. " O'Reilly Media, Inc."

Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S. and Tzoumas, K., 2015. Apache flink: Stream and batch processing in a single engine. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 36(4).

Chai F., (2013) Apache Hive: Data Warehousing and Analytics on Hadoop, Available at: https://15799.courses.cs.cmu.edu/fall2013/static/slides/hive.pdf, (Accessed: 30 March 2022)

Chambers, B. and Zaharia, M., 2018. Spark: The definitive guide: Big data processing made simple. " O'Reilly Media, Inc."

Drabas, T. and Lee, D., 2017. Learning PySpark. Packt Publishing Ltd.

Foote k., (2017), A Brief History of Big Data, Available at: https://www.dataversity.net/brief-history-big-data/ (Accessed: 9 Nov 2022)

IBM (2021), Connection State As Known by TCP, Available at: https://www.ibm.com/docs/en/zvm/7.1?topic=state-connection-as-known-by-tcp (Accessed: 27 Oct, 2022)

Lakshay A. (2020), Getting Started with Apache Hive – A Must Know Tool For all Big Data and Data Engineering Professionals, Available at: https://www.analyticsvidhya.com/blog/2020/10/getting-started-with-apache-hive/ (Accessed: 10 Nov 2022)

Moustafa, N. and Slay, J., 2015, November. 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). IEEE.

Northcutt S., Zeltser L., Winters S., Kent K., Ritchey R., (2005), Inside Network Perimeter Security, 2nd Edition, 2022 Pearson Education, Informit. 221 River Street, Hoboken, NJ 07030

Parkash, O., Machine Learning Integrated Big Data and High Performance Computing using Apache Storm.

Sedkaoui, S., (2018). Data analytics and big data. John Wiley & Sons.

Taylor D., (2022), Hive Tutorials for Beginners, Available at: https://www.guru99.com/hive-tutorials.html, (Accessed: 29 March 2022)

Theobald, O., 2017. Machine learning for absolute beginners: a plain English introduction (Vol. 157). Scatterplot press.

White, T., 2012. Hadoop: The definitive guide. " O'Reilly Media, Inc.".

Zoghi, Z. and Serpen, G., 2021. Unsw-nb15 computer security dataset: Analysis through visualization. arXiv preprint arXiv:2101.05067.