reference to system time $T \in t > 0$.

$S = \{S \mid S \in (0 \mid 1) System\ States\}$

$$S = \begin{cases} 0\ for,\ t = 0 \\ (0 \mid 1)\ for\ ,t > 0 \end{cases}$$

B. System Input: Let set I be defined as the input to the system. The inputs $ix, iy \in I$ are two input sources to the system.

$I = \{ix, iy\ are\ InputEvents\ of\ system\}$

$$I = \begin{cases} ix = packet\ \ data\ \ from\ network \\ iy = accessLogs\ \ from\ network \end{cases}$$

C. System Output: Let set O be defined as the output from the NIDS system. The inputs $ox, oy \in O$ are two output sources to the system.

$O = \{ox, oy \cup S^{+} \cup F\ OutputEvents\ with\ FinalState\}$

$$O = \begin{cases} ox = System\ Output \\ oy = Final\ \ State\ Output \end{cases}$$

D. Transition Function: Let set δ be defined as a Transition Function of the system. The system transaction depends upon system initial state S and with system functionality represented by δ.

$\delta = \sum (S^{+} * A * F)\ w.r.t\ Time t > 0\ of\ the\ system.$

$\delta = \{I_t * f_{id} * A_t\ w.r.t\ Time\ t > 0$

$f_{(1)} = Packet\ \ Analyzer(packet)$

$f_{(2)} = TrafficIngesion(dataset)$

$f_{(3)} = IntrusionDetection(packet, dataset)$

$f_{(4)} = IntrusionAnalysis(dataset)$

E. Final State: Let set F be defined as the final state of the system.

$F = \{z \in F \mid F\ is\ FinalState\ of\ system\}$

$$F = \begin{cases} z = 0\ \ System\ \ exit\ \ Successfully \\ z = 1\ System\ \ exit\ \ Unsuccessfully \end{cases}$$

A. System Model Diagram: The proposed NIDS model is represented and shows input to the system from various networks. The system includes two Hadoop and GPU modules to perform all system functions and generates output from the system. The operational flow of the system is represents in Fig.4.

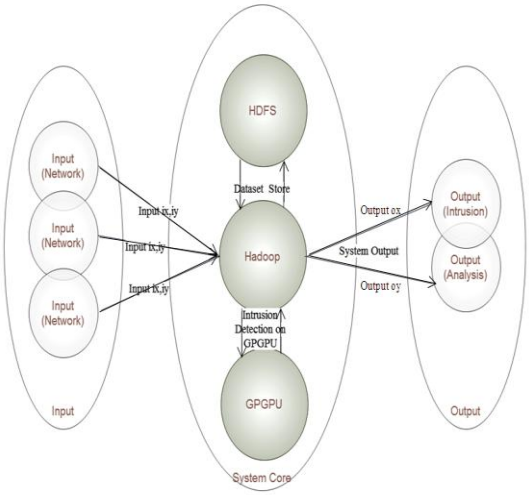

Figure 4: NIDS Proposed Model

7. Analysis

The proof of concept (POC) for the proposed systems is described below. We have implemented small functional module for Traffic Ingestion, Packet Analysis, Hadoop computation on log dataset, and data analytics over dataset. Depending on sample module and mathematical analysis of the following function, it is easy to understand the concept of integrating GPGPU into Hadoop framework. We have constructed a cluster of three computer systems of Pentium core i5 processor for observing preliminary outputs of system. A master node that has a Namenode and Tasktracker configure with 4 GBytes of RAM. Clusters are installed with Apache Hadoop 2.1.0 on Ubuntu 12.04 LTS operation system. Software's are Java 1.7 JDK, Eclipse Luna, Hadoop 1.0 plugins, HBase and Flume to be installed on a master node. Nodes are connected by 100 MBPS Ethernet cable.

7.1 Packet Analysis:

The packet analysis is carried out by using Python-Scapy packet crafting library [13]. Packet capturing script is written in python which provides fast packet designing, interactive packet and result manipulation.

A. Data Analytic using PF-ICF:

Using Hadoop, we get flexibility to perform wide range of analytics on Big-data. Here in our proposed system (NIDS), we applied PF-ICF to perform analysis on intrusions pattern over entire infrastructure. The data analysis requires some unique scoring values. In this approach we use scoring weight for intrusion pattern over the network. Depending upon pattern frequency over network cluster frequency, we can determine the wide spread of intrusions over the network.

B. Intrusions Types on the network:

a. Node 1 : DOS, DOS, ArpPoi, Tcp_Syn, Tcp_Syn, Tcp_Syn.
b. Node 2 : DOS, DOS, DOS, Tcp_Syn, Tcp_Syn, Tcp_Syn.
c. Node 3 : DOS, DOS, ArpPoi, Tcp_Syn, Tcp_Syn, Tcp_Syn.

TABLE I. PF-ICF Score Weight

Id	**Intrusion Detected on Network**		
	Denial of Services	**Tcp_Syn**	**ArpPoi**
Node 1(PF-ICF):	0.33	0.50	0.1872

Id	Intrusion Detected on Network		
	Denial of Services	Tcp_Syn	ArpPoi
Node 2(PF-ICF):	0.50	0.50	0
Node 3(PF-ICF):	0.33	0.66	0.1872

7.2 INTRUSION Statistics using PF-ICF:

The graph represents the maximum and minimum numbers of the vulnerabilities that are found across infrastructure. The above shown statistics of intrusions on network help a network administrator to configure the network security.

7.3 Observations:

a. If an intrusion appears on few clusters: Low Score.
b. If an intrusion appears on many clusters: High Score.

1. Ethernet Header :--> Destination MAC : a4:1f:72:8e:05:56, Source MAC : a4:1f:72:8e:00:92, Protocol : 8
2. IP Header =, IP Version : 4, Header Length : 5, TTL : 64, Protocol : 6, Source Address : 172.25.24.123, Destination Address : 172.25.24.124
3. TCP Header =, Source Port : 1500, Dest Port : 58171, Sequence Number : 279799081, Acknowledgement : 3489936958, TCP header length : 8
4. Data =, t 316:07:20 User: Sent Data with sig:E#^0z1#^! 11515115continue transition on DFA

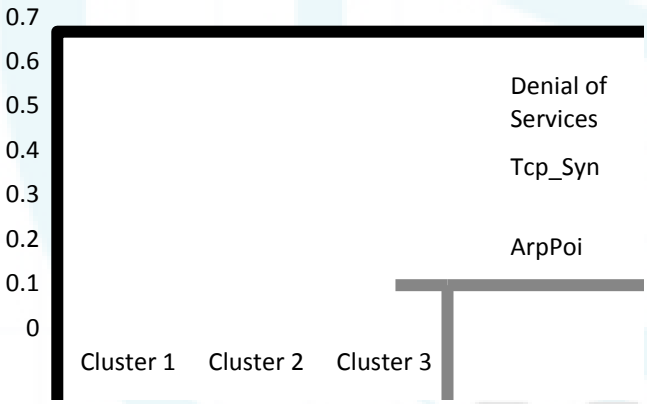

Figure 5: Intrusion Analytic using IP-ICF

8. Conclusion

In this paper, we have analyzed the design consideration of NIDS using Hadoop and GPGPU. The objective is to optimize NIDS performance by offloading intrusion mapping functionality from Hadoop to GP-GPU. In our proposed work, we have configured NIDS with Hadoop data platform in order to process large volumes of network traffic. We found that our design consideration is capable of processing log files of 1, 2, and 4 gigabytes in a very efficient time that of 29.86, 47.09, and 94.96 Seconds. Integration of Hadoop Data platform and NVidia GPGPU technology for offloading of pattern mapping job from Hadoop framework would result in optimizing NIDS performance.

In future work, the proposed NIDS system would capable to perform on high end computation based on Kepler GPGPU architecture in order to accelerate overall performance. The system is also very open to adapt high data analytic computation using Hadoop Eco-systems. The network security always remains a very challenging aspect so we must always keep on updating network security by applying new technology and approach.

References

[1] Axelsson, Stefan, "Intrusion Detection Systems: A Taxonomy and Survey", Technical Report No 99-15, Dept. of Computer Engineering, Chalmers University of Technology, Sweden, March 2000.
[2] H.Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of Intrusion Detection Systems", The International Journal of Computer and Telecommunications Networking - Special issue on computer network security, Volume 31 Issue 9, Pages 805 – 822, April 23, 1999.
[3] Holtz, Marcelo D, Bernardo David, Sousa Jr., R. T., "Building Scalable Distributed Intrusion Detection Systems Based on the MapReduce Framework", Telecomunicacoes (Santa Rita do Sapucai), v. 13, p. 22-31, 2011.
[4] Snort Official Page, Online 2014, Available: http://www.snort.org.
[5] Suricata Open Source IDS-IPS engine, Online 2014, Available: http://suricata-ids.org /2013/07/26/suricata-1-4-5-released/
[6] Apache Hadoop 2.4.1, Online 2014, Available:http://hadoop.apache.org/
[7] About GPGPU, Online 2014, Available: http://gpgpu.org/about.
[8] TESLA GPU ACCELERATORS FORSERVERS, Online 2014, http ://www.nVidia.com /object/tesla-servers.html.
[9] JeongJin Cheon, "Distributed Processing of Snort Alert Log using Hadoop", Department of Computer Engineering, Kumoh National Institute of Technology, Gumi, Gyeongbuk, Korea.
[10] Manish Kumar, "Scalable Intrusion Detection Systems Log Analysis using Cloud Computing Infrastructure", M. S. Ramaiah Institute of Technology, Bangalore and Research Scholar, Department of Computer Science and Applications, Bangalore University, Bangalore, INDIA.
[11] Prathibha,"Design of a Hybrid Intrusion Detection System using Snort and Hadoop", International Journal of Computer Applications (0975 – 8887), Volume 73– No.10, July 2013.
[12] Puneet Goswami, PhD,"The DF-ICF Algorithm- Modified TF-IDF", International Journal of Computer Applications (0975 – 8887), Volume 93 – No.13, May 2014, Associate Professor Galaxy Global Group of Institutions Dinarpur Ambala, Haryana, India.
[13] About Scapy, Online 2014, http://www.secdev.org/ projects/scapy/.
[14] Herodotos Herodotou,"Hadoop Performance Models", Technical Report, CS-2011-05, Computer Science Department, Duke University.
[15] Chen-Hsiung Liu, "PFAC Library GPU-based string matching algorithm", Online 2014, http://code.google.com/p/pfac/