**Novel Architecture: anomaly detection in Multi aspects streams**

**Abstract**

Intrusion Detection Systems provide protection for network systems and can be categorized as either anomaly based, or misuse based or collaboration of both. This paper proposes a novel architecture as a solution to Network Intrusion Detection Systems. Associated IDS have utilized the KDD99 dataset, which is limited ibn its variety of attacks analysis, the representation analysis of the proposed analysis is based on the UNSW\_NB15 dataset.

**Keywords**: Intrusion detection System, Network Intrusion Detection System, Anomaly based intrusion, Attack detection rate, false alarm rate, misuse-based intrusion.

1. **Introduction**

Attacks are set of occurrences that trade-off the fundamentals of computer systems [6], such as availability, confidentiality, and integrity of the network system. Firewall systems have limitations in detecting present-day attacks and incapable of scrutiny of network packets in depth, hence the need for IDSs which are equipped of cyber security infrastructure.

Network Intrusion Detect System (NIDS) detects network traffic and determines if a network packet is an attack or a normal traffic, NIDS can be categorized into [6] misuse/signature and anomaly based. The signature based applies known attacks as signature and patterns to detect commands, while anomaly-based implements detection through a normal profile generated from the normal conduct of the network, any straying from the normalcy will be taken as an attack or anomaly.

Misuse/Signature based intrusion detection application [4] can genuinely establish intrusion attacks in connection to the known signature attacks, the challenge with this arrangement is when it is faced by unknown intrusions which then leads to dawning mediation of security experts to define new set of rules required to detect such attacks.

Anomaly-based IDS implements intrusion detection through models by training the model to recognize the pattern of the attacks. The system [4] builds the model for normal intrusion, hence when new data from the network is received as input, any variation from normalcy is regarded as attack, this sometimes leads to many false positive.

In recent times IDSs have evolved into two classes [5] namely (i) Network Intrusion Detection Systems (NIDS); they usually administer on network traffic; this method of detection is carried out by Firewalls. The second class of IDS is Host Intrusion Detection Systems (HIDS), they are stationed on each network host, they use data collated from the network traffic to detect intrusion.

Technological advancement led audit logs to be moved online and many other programs used to analyze data, auditing data logs usually consumed the bandwidth of the network which led the administrators to audit the data logs at night when systems user load was low. This approach also meant that attacks would have occurred well in advance before it was detected. Real time Intrusion Detection Systems was developed in the early 90’s enabling attacks to be detected as well as attack preemption [9].

The resolution to past deficiencies of NIDs brought about anomaly-based Network Intrusion Detection Systems (ABNID). ABNIDs are systems which implement Artificial intelligence, using machine learning techniques to develop models to detect attacks without patterns [5].

ABNIDs can be categorized into the following:

1. Statistical anomaly detection: This involves functioning models [5], threshold metric, mean and standard deviation, multivariate models, and Time series models. It accumulates data from normal traffic and institutes a pattern, it then periodically illustrates the network occurrences based on statistical models and likens it to the already established pattern, the IDS the reports any deviance to the network administrator.
2. Data Mining based detection: This includes methods such as Clustering, association rule discovery, and Classification [5]. This detection method involves the use of IDS designers automated tools and techniques analyze audit data thereby drawing out features of an attack, this process leads to computation of rules that is used to [9] discover attacks. The IDS designer would be required to label the audit data that is associated with the misuse. The duty of labelling the misuse data can also be automated.
3. Machine Learning based Detection: This comprises of Bayesian models, Neural networks, Fuzzy logic models and Support Vector Machine [5]. It probes the assembly of algorithms that can learn from and make predictions on input data [10]. Machine learning can be grouped as supervised learning when the training data is labeled, unsupervised when the training data is unlabeled and semi-supervised when the training dataset is a mix of labeled and unlabeled dataset.
4. Knowledge based detection: This consist of State transition analysis, Expert systems, and Signature analysis [5]. This comprises of the ontology, knowledge-base and the reasoning logic. The [11] ontology has three classes namely means, consequences and targets. The means class compresses the way and method of an attack, consequences class compresses the outcomes of an attack, and the target class compresses the information oof the system under attack [11]. The knowledge-base is constructed by encrypting the input data from different channels as OWL (Web Ontology language) which is then converted to N3 triples (format for storing and transmitting data). The entities collected from the different data sources can be assets of a class in the ontology which could include such properties as IP address, port number etc. The reasoning logic module takes input [11] from different data streams, the knowledge declared in the knowledge base, and the ontology to determine if an attack occur or if the input data is a normal traffic.
5. **Background and Previous work**

A search of the literature discloses some academic work on Network Intrusion Detection, Vikash et al. applied an integrated rule based to Intrusion Detection [1], the NIDS scans the approaching traffic in real time and on a fast connection. Different decision tree models such as C5, CHAID, CART, QUEST is trained with selected 123 features of the dataset. The Rile based is generated from the diverse models, the rules chosen from each model is based on their threshold confidence factor. The confidence factor of a leaf node is the number of instances genuinely classified [1] by the leaf to the total number of instances classifies by that leaf. The model was developed by using the UNSW-NB15 dataset, rules are chosen for each attack from each model using their confidence factor thereby applying the pattern to diagnose incoming traffic based on the rules developed from confidence factor of each model learned.

Siddharth et al. [2] proposed a Microcluster-based detector of anomalies in Edge Streams, this approach involves the streaming hypothesis testing approach, detection and guarantees and incorporating relations. The hypothesis framework requires a restrictive Gaussian assumption which can lead to excessive false positive or negative, a weak assumption is made that the mean level in the current time tick is the same as the mean level before the current time tick, this avoids a strict assumption of stationary over time.

The proposed algorithm equips streaming hypothesis testing approach which uses streaming data structures withing a hypothesis testing-based structure that assures a false positive probability. is assumed to be the total number of edges from to up to the current time, is assumed as the number of edges from to in the current time tick, the chi-squared statistic is as follows [2]:

The streaming anomaly scoring is given as

Where CMS is the count mean sketch.

In another approach taking by Siddharth et al. [3] for fast anomaly detection in Multi-aspect streams utilized the hash functions. The hash functions include the FEATUREHASH and RECORDHASH systems, the FEATUREHASH consist of hash functions that is applied a single feature [3]. In categorical data (e.g. IP address), a standard hash function is used while in real-valued data (e.g. average packet length) a streaming log-bucketization system is utilized. The process involves collecting the processed data from the log-transform then performing a min-max normalization of the data, the data is then mapped into buckets. The second approach of RECORDHASH, the entire record is divided into two, one part for the categorical features and the other part for the real-valued features. The categorical data is hash into bucket and the real valued data is hash into buckets, the sum modulo of the categorical bucket and the real-valued bucket is the calculated. The essence of this approach to proffer solution to intrusion detection in large data streams, thus approach considers the problem addressing anomalies in multiple streaming platforms. The use of locality-sensitive hash functions which hash input data into fixed number of buckets, hence many related input data are hashed into related buckets. Separating the input data into similar buckets is done using temporal scoring approach that considers the how much overlap is spotted between buckets at any time, a high percentage of overlap over a short period of time indicates the existence of anomalous activity.

Sumit et al. [11] proposed a knowledge-based approach to intrusion modeling with its architecture divided into two sections namely data streams and ontology knowledge-base and reasoner. The data streams comprise different channels that supply useful information related to an attack, any malicious activity from the streams is declared in the knowledge-base. The knowledge-base module consists of the [11] ontology, knowledge-base, and the reasoning logic. The ontology has three basics: ‘means’ class which summarizes the routines of an attack, the ‘consequences’ class summarizes the aftermath of the attack and ‘target’ which summarizes the information of the system under attack. The reasoning logic takes input from different data streams and the learning inferred from the streams is declared to the knowledge base and ontology to detect the possibility of an attack.

**3. Proposed Work**

**3.1 Data Set Description**

The raw network packets of the UNSW-NB 15 dataset were created by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviors.

Tcpdump tool is utilized to capture 100 GB of the raw traffic which are Pcap files. This dataset has nine types of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. The Argus, Bro-IDS tools are used, and twelve algorithms are developed to totally generate 49 features with the class label.

The features of the dataset are displayed below:

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | |  |  |  | | --- | --- | --- | | srcip |  |  | | nominal | Source IP address |
| 2 | |  |  |  | | --- | --- | --- | | sport |  |  | | integer | Source port number |
| 3 | |  |  |  | | --- | --- | --- | | dstip |  |  | | nominal | Destination IP address |
| 4 | Dsport | Integer | Destination port number |
| 5 | Proto | Nominal | Transaction protocol |
| 6 | State | Nominal | Indicates the state and its dependent protocol |
| 7 | Dur | Float | Record total duration |
| 8 | Sbytes | Integer | Source to destination transaction bytes |
| 9 | Dbytes | Integer | Destination to source transaction bytes |
| 10 | Sttl | Integer | Source to destination time to live value |
| 11 | Dttl | Integer | Destination to source time to live value |
| 12 | Sloss | Integer | Source packets retransmitted or dropped |
| 13 | Dloss | Integer | destination packets transmitted or dropped |
| 14 | Service | Nominal | http, ftp, smtp, ssh, dns, and (-) if not much used service |
| 15 | Sload | Float | Source bit per second |
| 16 | Dload | Float | Destination bit per second |
| 17 | Spkts | Integer | Source to destination packet count |
| 18 | Dpkts | Integer | Destination to source packet count |
| 19 | Swin | Integer | Source TCP window advertisement value |
| 20 | Dwin | Integer | Destination TCP window advertisement value |
| 21 | Stcpb | Integer | Source TCP base sequence number |
| 22 | Dtcpb | Integer | Destination TCP base sequence number |
| 23 | Smeansz | Integer | Mean of the packet size transmitted by the source |
| 24 | Dmeanz | Integer | Mean of the packet size transmitted by the destination |
| 25 | Trans\_depth | Integer | Represents the pipeline depth into the connection of http request/response transaction |
| 26 | Res\_bdy\_len | Integer | Actual uncompressed content size of the data transferred from the server’s http service |
| 27 | Sjit | Float | Source jitter (msec) |
| 28 | Djit | Float | Destination jitter (msec) |
| 29 | Stime | Timestamp | Record start time |
| 30 | Ltime | Timestamp | Record last time |
| 31 | Sintpkt | Float | Source interpacket arrival time (msec) |
| 32 | Dintpkt | Float | Destination interpacket arrival time (msec) |
| 33 | Tcprtt | Float | TCP connection setup round trip time, the sum of ‘synack’ and ‘ackdat’. |
| 34 | Synack | Float | TCP connection set up time, the time between the SYN and the SYN\_ACK packets. |
| 35 | Ackdat | Float | TCP connection setup time, time between the SYN\_ACK and the ACK packets. |
| 36 | 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 |
| 37 | Ct\_state\_ttl | Integer | No. for each state (6) according to specific range of values for source/destination time to live (10) (11) |
| 38 | Ct\_flw\_http\_mthd | Integer | No. of flows that has methods such as Get and Post in http service |
| 39 | Is\_ftp\_\_login | Binary | If the ftp session is accessed by user and password, then 1 else 0 |
| 40 | Ct\_ftp\_cmd | Integer | No of flows that has a command in ftp session |
| 41 | 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) |
| 42 | 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) |
| 43 | Ct\_dst\_ltm | Integer | No of connections of the same destination address (3) in 100 connections according to the last time (260) |
| 44 | Ct\_src\_ltm | Integer | No of connections of the same source address (1) in 100 connections according to the last time (26) |
| 45 | 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) |
| 46 | 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) |
| 47 | Ct\_dst\_src\_sport\_ltm | Integer | No of connections of the same source (1) and the destination (3) address in 100 connections according to the last time (26) |
| 48 | Attack\_cat | Nominal | The name of each attack category. |
| 49 | Label | Binary | 0 for normal and 1 for attack records |

Below is the description of [5] attack types for the UNSW-NB15 dataset:

|  |  |
| --- | --- |
| Traffic Type | Description |
| Analysis | Generic type for describing port scanning, spam, and html file penetration. |
| Backdoor | Malware type that negates normal authentication to grant remote access to resources. |
| DOS | Deprives legitimate users from using web services through flooding the network with invalid authentication attempts forcing it to crash. |
| Exploits | Often incorporated into malware, allowing an easy and fast breeding. |
| Fuzzing | Automated process of finding software vulnerability by randomly feeding different arrangement of data into a target program until the program vulnerability is revealed. |
| Generic | Collision attack on the secret keys of ciphers, works against all block ciphers. |
| Normal | Traffic that contains no threat. |
| Reconnaissance | The assembly of simple methods that gather information about the target network. |
| Shellcode | Set of instructions which are introduced and executed by a flawed program, and it exploits registers and the function of a program. |
| Worms | It consumes too much system memory and network bandwidth thereby diminishes the system availability. |

The efficacy of the NIDS is dependent on the robustness of the intrusion dataset that the algorithm is trained on to identify attacks. Previous datasets used in NIDS research such as KDD98, KDDCUP 99, NSLKDD, ADFA, and DARPA have their limitations which renders them invaluable in recent NIDS research because a lot of attacks have been developed during development in cyber-attacks. Due to these concerns the Australian Center for Cyber Security in collaboration with researchers around the world created the UNSW-NB15 dataset. To accomplish the task of developing an intrusion dataset that will stand the test of time in aiding modeling of NIDS made use of IXIA Perfect storm tool to produce a rich hybrid set of normal and abnormal modern network traffic [5]. The UNSW-NB15 strives to imitate modern network environments by implementing more recent and modern attack patterns, the attack types covered in the dataset include Normal, Fuzzing, Analysis, Backdoor, DOS, Exploits, Generic, Reconnaissance, and worms.

Below is a table showing the attack data distribution in the training [5] and testing data of the UNSW-NB15.

|  |  |  |
| --- | --- | --- |
| Category | Training set | Testing set |
| Normal | 56,000 | 37,000 |
| Analysis | 2,000 | 677 |
| Backdoor | 1,746 | 583 |
| DOS | 12,264 | 4,089 |
| Exploits | 33,393 | 11,132 |
| Fuzzing | 18,184 | 6,062 |
| Generic | 40,000 | 18,871 |
| Reconnaissance | 10,491 | 3,496 |
| Shellcode | 1,133 | 378 |
| Worms | 130 | 44 |
| Total Records | 175,341 | 82,332 |

**4.Novel Architecture and Implementation**

**5. Results and discussion**

**6. Conclusion and Recommendation**

**References**

[1] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, “An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset,” *Cluster Comput*, vol. 23, no. 2, pp. 1397–1418, Jun. 2020, doi: [10.1007/s10586-019-03008-x](https://doi.org/10.1007/s10586-019-03008-x).

[2] S. Bhatia, B. Hooi, M. Yoon, K. Shin, and C. Faloutsos, “MIDAS: Microcluster-Based Detector of Anomalies in Edge Streams,” *arXiv:1911.04464 [cs]*, Aug. 2020, Accessed: Sep. 14, 2021. [Online]. Available: <http://arxiv.org/abs/1911.04464>

[3] S. Bhatia, A. Jain, P. Li, R. Kumar, and B. Hooi, “MStream: Fast Anomaly Detection in Multi-Aspect Streams,” in *Proceedings of the Web Conference 2021*, Ljubljana Slovenia, Apr. 2021, pp. 3371–3382. doi: [10.1145/3442381.3450023](https://doi.org/10.1145/3442381.3450023).

[4] M. H. Aghdam and P. Kabiri, “Feature Selection for Intrusion Detection System Using Ant Colony Optimization,” p. 13, 2016.

[5] S. Meftah, T. Rachidi, and N. Assem, “Network Based Intrusion Detection Using the UNSW-NB15 Dataset,” p. 11.

[6] N. Moustafa and J. Slay, “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)*, Canberra, Australia, Nov. 2015, pp. 1–6. doi: [10.1109/MilCIS.2015.7348942](https://doi.org/10.1109/MilCIS.2015.7348942).

[7] R. Abdulhammed, M. Faezipour, and K. M. Elleithy, “Network intrusion detection using hardware techniques: A review,” in *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, Apr. 2016, pp. 1–7. doi: [10.1109/LISAT.2016.7494100](https://doi.org/10.1109/LISAT.2016.7494100).

[8] O. Osho and S. Hong, “An Overview: Stochastic Gradient Descent Classifier, Linear Discriminant Analysis, Deep Learning and Naive Bayes Classifier Approaches to Network Intrusion Detection,” *International Journal of Engineering Research*, vol. 10, no. 04, p. 15.

[9] S. Stolfo, W. Lee, P. Chan, W. Fan, and E. Eskin, “Data Mining-based Intrusion Detectors: An Overview of the Columbia IDS Project.,” *SIGMOD Record*, vol. 30, pp. 5–14, Dec. 2001.

[10] O. G. Abbas, K. Khorzom, and M. Assora, “Machine Learning based Intrusion Detection System for Software Defined Networks,” *International Journal of Engineering Research & Technology*, vol. 9, no. 9, Oct. 2020, Accessed: Sep. 14, 2021. [Online]. Available: [https://www.ijert.org/research/machine-learning-based-intrusion-detection-system-for-software-defined-networks-IJERTV9IS090508.pdf, https://www.ijert.org/machine-learning-based-intrusion-detection-system-for-software-defined-networks](https://www.ijert.org/research/machine-learning-based-intrusion-detection-system-for-software-defined-networks-IJERTV9IS090508.pdf,%20https:/www.ijert.org/machine-learning-based-intrusion-detection-system-for-software-defined-networks)

[11] S. More, M. Matthews, A. Joshi, and T. Finin, “A Knowledge-Based Approach to Intrusion Detection Modeling,” in *2012 IEEE Symposium on Security and Privacy Workshops*, San Francisco, CA, USA, May 2012, pp. 75–81. doi: [10.1109/SPW.2012.26](https://doi.org/10.1109/SPW.2012.26).