World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

A network intrusion prediction model using Bayesian network

Ijegwa David Acheme [1,*], Adebanjo Adeshina Wasiu [2] and Glory Nosa Edegbe [1]

[1] *Department of Computer Science, Edo State University, Uzairue, Nigeria.*
[2] *Department of ICT Education, Africa Center of Excellence for Innovative and Transformation STEM Education (ACEITSE), Lagos State University, Lagos.*

World Journal of Advanced Research and Reviews, 2024, 23(01), 2813–2821

Publication history: Received on 11 June 2024; revised on 25 July 2024; accepted on 27 July 2024

Article DOI: https://doi.org/10.30574/wjarr.2024.23.1.2155

Abstract

Intrusion detection systems are increasingly becoming more and more useful in the fight against cyber threats. As businesses continue to transition into adopting information processing systems including cloud computing, there is a greater need for the development of safety measures to ensure the preservation of information against theft, unauthorized access and other cyber threats. Several articles in literature have reported the development of intrusion detection and prediction system using varying techniques. In this research work, a simple Bayesian model is presented. To build this model, the widely used dataset of NSL-KDD was used. Existing literature was relied on for the selection of the best features while the Max-Min Hill Climbing (MMHC) algorithm was used to define the Bayesian network structure. The model was then implemented using the Genie Bayesian Modelling software. Results of simulation, influence and scenario analysis established the efficacy of the model in predicting the likelihood of an intrusion in a network environment.

Keywords: Intrusion detection; Bayesian Network; Cyber security; Machine Learning; Information Systems

1. Introduction

Intrusion detection and prediction Systems are very important components of modern cybersecurity frameworks. They are mainly designed to monitor, detect, and respond to unauthorized or malicious activities within a computer network or system. These systems operate by analyzing network traffic, system logs, and other data sources to identify potential security breaches or suspicious behavior that may indicate an attack or compromise. IDS are broadly categorized into two main types, these are; Network-based Intrusion Detection Systems (NIDS) and Host-based Intrusion Detection Systems (HIDS) The former are designed to monitor network traffic for suspicious activity by analyzing packet data as it traverses network segments, while the later are systems that operate on individual hosts or devices, analyzing the device's operating system and application logs for signs of malicious activity (Kumar, et al. (2021); Makinde & Acheme, (2023)).

In this continuously changing landscape of cybersecurity, IDS plays vital role in protecting information systems from a many types of threats. They provide early warning signals of potential security breaches, enabling organizations to respond swiftly to mitigate damage. IDS also assures protection of sensitive data by detecting unauthorized access attempts, thereby safeguarding sensitive data from being stolen, altered, or destroyed. Furthermore, IDS offer valuable insights into network activity, helping security teams understand the nature and scope of threats, which can inform future defenses.

The cyber threat landscape is becoming increasingly complex, driven by several factors including Sophistication of Attacks, Proliferation of Attack Vectors, Automation and AI in Cybercrime and Insider Threats. The rise of new

* Corresponding author: Ijegwa David Acheme

technologies and platforms, such as cloud computing, Internet of Things (IoT), and mobile devices, has even more expanded potential attack resources, providing more entry points for attackers. Attackers are using automation and artificial intelligence to launch large-scale, coordinated attacks that can adapt and evolve in real time. The risk from insiders, whether malicious or negligent, adds another layer of complexity, requiring systems to monitor internal activities closely.

Given the evolving threat landscape, traditional IDS methods are facing significant challenges, ranging from High False Positive Rates, Detection of Unknown threats and scalability issues. To address these challenges, advanced detection methods, such as the integration of machine learning and probabilistic models like Bayesian networks, are becoming essential. These methods offer the advantages of improved accuracy since the machine learning algorithms can learn from vast amounts of data, improving the accuracy of threat detection by identifying subtle patterns indicative of malicious behavior. They are also adaptable offering flexible frameworks for modelling uncertainties as well as providing real-time analysis and timely alerts to enable quicker response actions

As cyber threats continue to grow in complexity, the need for sophisticated and adaptive intrusion detection systems is more critical than ever. Several research works have reported the use of machine learning modelling for building IDS. Various Intrusion Detection Systems (IDS) have been documented in the literature, employing different techniques to monitor and detect unauthorized activities or anomalies within a network or system. These techniques can be broadly categorized into signature-based detection, anomaly-based detection, and hybrid detection methods. Each approach has its own strengths and weaknesses, making them suitable for different scenarios and types of threats.

A Bayesian network, also known as a Bayesian belief network or probabilistic directed acyclic graphical model, is a statistical model that represents a set of variables and their conditional dependencies using a directed acyclic graph (DAG) (Ijegwa et al., 2019). Bayesian networks are utilized for probabilistic inference, enabling the computation of the likelihood of various outcomes given certain conditions or evidence. They are particularly effective for handling uncertainty and incorporating prior knowledge into the analysis.

Bayesian Networks (BNs) offer a seamless convergence of Artificial Intelligence (AI) and Statistics. As part of the family of probabilistic graphical models, they aim to build models from data and, at times, from expert opinion. The application of Bayesian Networks spans tasks such as time series prediction, decision-making under uncertainty, and anomaly detection. A BN consists of three main components: nodes, associated conditional probabilities, and links between the nodes. The nodes represent the variables, while the links illustrate influence and dependency. Table 1 summarizes these components of a Bayesian Network system.

Table 1 Components of a Bayesian Network

Component	**Definition**	**Example**
Nodes (Vertices)	Nodes in a Bayesian network represent random variables. Each node corresponds to a specific variable that can take on different states or values.	In an intrusion detection context, nodes might represent variables such as "Network Traffic Volume," "Number of Failed Login Attempts," "Presence of Malware," etc.
Edges (Directed Arows)	Edges are directed arrows that connect pairs of nodes, representing the conditional dependencies between the variables. An edge from node A to node B indicates that A directly influences B	The direction of the arrow signifies the dependency, where the parent node influences the child node. For example, an edge from "Presence of Malware" to "System Performance" indicates that the presence of malware affects system performance.
Conditional Probability Tables (CPTs)	Conditional Probability Tables (CPTs) quantify the relationships between connected nodes in a Bayesian Network. Each node has an associated CPT that defines the probability distribution of that node based on the states of its parent nodes.	For a node X with parents P_1, P_2, P_3……P_n, the CPT defines the probability $P(X/P_1 , P_2 , P_3……P_n)$. Suppose a node "Alert" is influenced by "Network Traffic Volume" and "Number of Failed Login Attempts." The CPT for "Alert" would list the probabilities of different alert states (e.g., high, medium, low) for each combination of states of the parent nodes

2. Related work

Saranya et al. (2020) reviewed various Machine Learning (ML) algorithms for Intrusion Detection Systems (IDS) across applications such as fog computing, the Internet of Things (IoT), big data, smart cities, and 5G networks. They classified intrusions using ML algorithms like Linear Discriminant Analysis (LDA), Classification and Regression Trees (CART), and Random Forest, tested on the KDD-CUP dataset. Their study compared the efficiency of these algorithms with recent research findings.

Ahmad et al. (2021) provided a taxonomy of significant Machine Learning and Deep Learning (DL) techniques used in network-based IDS (NIDS) systems. They discussed the strengths and limitations of these solutions, current trends, advancements, and highlighted research challenges and future directions, focusing on methodologies, evaluation metrics, and dataset selection.

Amouri et al. (2020) proposed a two-stage IDS: the first stage collects data through dedicated sniffers (DSs) and generates CCI, which are sent to a super node (SN). In the second stage, the SN uses linear regression on the collected CCIs to distinguish between benign and malicious nodes. The system's detection performance was evaluated in extreme network scenarios involving varying power levels and node velocities for Random Way Point (RWP) and Gauss Markov (GM) mobility models. The IDS achieved detection rates above 98% in high power/node velocity scenarios and around 90% in low power/node velocity scenarios, focusing on detecting blackhole and DDoS attacks.

Verma & Ranga (2020) explored machine learning classification algorithms for defending IoT against DoS attacks. They conducted a comprehensive study of classifiers to enhance anomaly-based IDS development, evaluating performance using prominent metrics and validation methods on CIDDS-001, UNSW-NB15, and NSL-KDD datasets. They also performed statistical analyses using Friedman and Nemenyi tests to identify significant differences among classifiers and evaluated the response time of classifiers on IoT hardware using Raspberry Pi. Their study aimed to encourage IoT security researchers to develop IDSs using ensemble learning and propose methods for statistically assessing classifier performance.

Jaradat et al. (2022) proposed a model for intrusion detection and classification using machine learning techniques. The model involved acquiring and transforming the dataset, performing feature selection, and processing the refined dataset with the Konstanz Information Miner (KNIME). They applied three classifiers for comparative analysis using the CICIDS2017 dataset. Experimental results showed accuracy rates ranging from 90.59% to 98.6%. This research demonstrated the potential of machine learning in cybersecurity and data analysis, encouraging the development of more accurate intrusion detection systems. Other applications of machine learning and Bayesian networks to solving contemporary societal issues are found in (Acheme et al, 2023; Acheme & Vincent, 2021; Acheme et al, 2020; Acheme et al, 2021; Acheme, & Enoyoze, 2024)

2.1. Background of Bayesian Network

Bayesian Networks (BNs) provide an effective integration of Artificial Intelligence (AI) and Statistics. They are a type of probabilistic graphical model designed to build models using data, and sometimes expert opinions. BNs are applied to various tasks including time series prediction, decision-making under uncertainty, and anomaly detection. A Bayesian Network consists of nodes, which represent variables, and links between nodes that indicate influence and dependency.

The general equation for a Bayesian Network is shown in Equation 1.1.

Let $X_1, X_2 \dots \dots \dots X_n$ be a set of random variables representing the nodes in the Bayesian network. The network can be denoted by a directed acyclic graph (DAG) G, where each node X_i represents a random variable, and each directed edge $(X_i \rightarrow X_j)$ represents a probabilistic dependency between X_i and X_j. The joint probability distribution of all the random variables in the Bayesian network can be expressed as the product of the conditional probability distributions of each variable given its parents.

$$P(X_1, X_2 \dots \dots, X_n) = \Pi P(X_i | parents(X_i)) \quad (1.1)$$

Where

P(X_i) is the conditional probability distribution of variable X_i

$parents(X_i)$ denotes the set of parent nodes of X_i in the DAG

The equation allows us to efficiently compute probabilities and perform inference in the Bayesian network by propagating information through the graph and using Baye's rule to update probabilities based on observed evidence.

A Bayesian network is fully specified by the combination of: The structural specification (nodes and connecting arcs) and the probability distribution $P(X_i|\prod x_i)$ attached to each node.

- Building a BN can follow the following steps:
- Identifying the variables (nodes)
- Defining the structural specification
- Feature selection to determine which variables are most likely to influence each other
- Finally assigning probability distributions to the nodes

Once a BN has been built as stated above, it can be used to compute any conditional probability (CPTs) one wishes to compute. They are very convenient for representing systems of probabilistic causal relationships. The fact "M" often leads or causes "N" can be modeled in the network by adding a directed arc from M to N and assigning the appropriate probabilities to the nodes.

3. Methodology

This work follows a methodology that is applicable to most machine learning and data science projects. It starts with data collection, data cleaning and pre-processing, feature selection, the bayesian network construction, and testing.

3.1. Data Collection

The dataset utilized in this research is the NSL-KDD dataset [NSL-KDD], which serves as a new standard for evaluating network intrusion detection systems. This dataset includes selected records from the original KDD 99 dataset and addresses its issues. The NSL-KDD dataset contains connection records with 41 features, of which 34 are numeric and 7 are symbolic or discrete. The training set of NSL-KDD includes 22 attack types, with an additional 17 types present only in the testing set. Table I provides a detailed description of the NSL-KDD dataset features.

Table 2 Features of the NSL-KDD Dataset

S/N	Name of Features	Description
1	Duration	Measure of time (in seconds) for which the connection lasted
2	Protocol_type	Type of connection protocol
3	service	Destination and type of service
4	Flag	Connection status flag
5	Src_bytes	Total number of data bytes utilized form source to destination
6	Dst_bytes	Total number of data bytes utilized from destination to source
7	Land	This is a binary output if connection from same port and 0 if not
8	Wrong_fragment	Total number of wrong fragments
9	Urgent	Total number of urgent packets
10	Hot	Total number of hot indicators
11	Failed_logins	Total number of failed login attempts
12	Logged_ins	This is a binary variable indicating successful login (1) or unsuccessful (0)
13	Num_compromised	Total number of breached/compromised conditions
14	Root_shell	This is a binary variable. 1 if the root shell is obtained or 0 if not
15	Su_attempted	This is a binary variable. 1 if the su_root command is attempted 0 if not
16	Num_root	This is the total number of "root" access

17	Num_file_creations	Total number file creation operations
18	Num_shells	Total number of shell prompts
19	Num_access_files	Total number of operations on access files
20	Number_outbound-cmds	Total number of ftp sessions' outbound command
21	Is_host_login	This is a binary variable. 1 if the login is from the host, 0 if not
22	Is_guest_login	This is a binary variable. 1 if the login is from a guest, 0 if not
23	Count	Total number of connection to same host.
24	Srv_count	Total number of connections to the same service as the current connection
25	Serror_rate	The percentage of connections that have "SYN" errors
26	Srv_serror_rate	The percentage of connections that have "SYN" err
27	Rerror_rate	The percentage of connections that have "REJ" err
28	Srv_rerror_rate	The percentage of connections that have "REJ" err
29	Same_srv_rate	The percentage of connections to the same service
30	Diff_srv_rate	The percentage of connections to different services
31	Srv_diff_hist_rate	The percentage of connections to different hosts
32	Dst_host_count	Total count of connections ending in the the same destination
33	Dst_host_srv_count	Total count of connections ending in the the same destination and using the same service
34	Dst_host_same_srv_rate	Percentage of connections with the same destination
35	Dst_host_diff_srv_rate	The percentage of different services running on the host
36	Dst_host_same_src_port_rate	The percentage of connections to current port having the same port
37	Dst_host_srv_dif_host_raye	The percentage of connections to the same service coming from different hosts
38	Dst_host_serror_rate	The percentage of connections to the current host that have an SO error
39	Dst_host_srv_serror_rate	The percentage of connections to the current hsot and specified service that have na SO error
40	Dst_host_rerror_rate	The percentage of connections to the current host with RST
41	Dst_host_srv_rerror_rate	The percentage of connections to the current host and specified service that have an RST

3.2. Data Preprocessing

Table 1 represents the features of the datasets collected. The data in this form cannot be used for Bayesian modelling, hence necessary preprocessing was required in order for the data to be usefull. Specifically, symbolic features were mapped to numeric variables. Also, in order to predict an attack, it was necessary to classify the attack names to "Attack" and "normal".

3.3. Feature Selection

In order to built the Bayesian model, relevant features needed to be identified, these features are represented as the nodes of the model, with the connecting links represents their dependence. From existing literature and domain knowledge, the key features relevant to intrusion detection were identified. Majority of these are network traffic variables, connection-related variables, user activity variables, system performance variables, and security-specific variables.

Further more, we used statistical methods to assess the correlation between features and their relevance to intrusion detection. Which was helpful in selecting the most significant features for the Bayesian network.

3.4. Bayesian Network Construction

The task of building the model is represented by figure 3.1. It captures the fur critical steps for constructing the bayesian model.

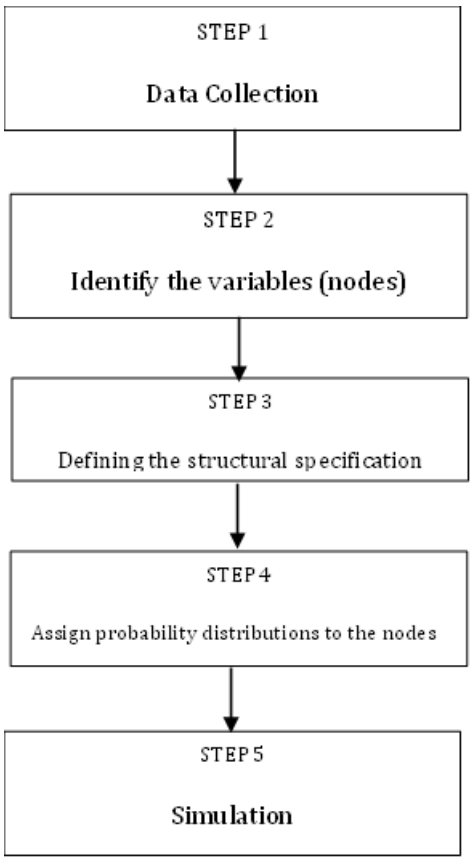

Figure 1 Flow chart representing the methodology

From figure 3.1, steps 1 and 2 have been shown in sections 3.1 to 3.4.

3.5. Structural Specification of the Model

When constructing a Bayesian network for an Intrusion Detection System (IDS), learning the structure from data is a critical step. The aim of the structure learning process to identify the dependencies between variables so as to establish a directed acyclic graph (DAG) that best represents their statistical dependencies and relationships. different algorithms have been reported in literature, these are broadly categorized into three types: constraint-based methods, score-based methods, and hybrid methods. This work makes use of the hybrid method shown below and represented by figure 3.2

3.6. Max-Min Hill Climbing (MMHC) Algorithm

Steps:

- Constraint-Based Phase: Use a constraint-based approach to identify an undirected skeleton of the network.
- Score-Based Phase: Orient the edges of the skeleton using a score-based search, (hill climbing).
- Advantages: Efficiently narrows down the search space using constraints before refining the structure with a scoring method.

```python
import pandas as pd
from pgmpy.estimators import PC, HillClimbSearch, BicScore

# Load dataset
data = pd.read_csv('nsl_kdd_dataset.csv')

# Constraint-Based Method: PC Algorithm
pc = PC(data)
skeleton = pc.build_skeleton()
pc_model = pc.estimate_cpds(skeleton)

# Score-Based Method: Hill Climbing
hc = HillClimbSearch(data, scoring_method=BicScore(data))
best_model = hc.estimate()

# Hybrid Method: Max-Min Hill Climbing
# Constraint phase: (using skeleton from PC algorithm)
skeleton = pc.build_skeleton()
mmhc = HillClimbSearch(data, scoring_method=BicScore(data))
hybrid_model = mmhc.estimate(start=skeleton)
```

Figure 2 Implementation of Max-Min Hill Climbing (MMHC) Algorithm

4. Implementation and Results

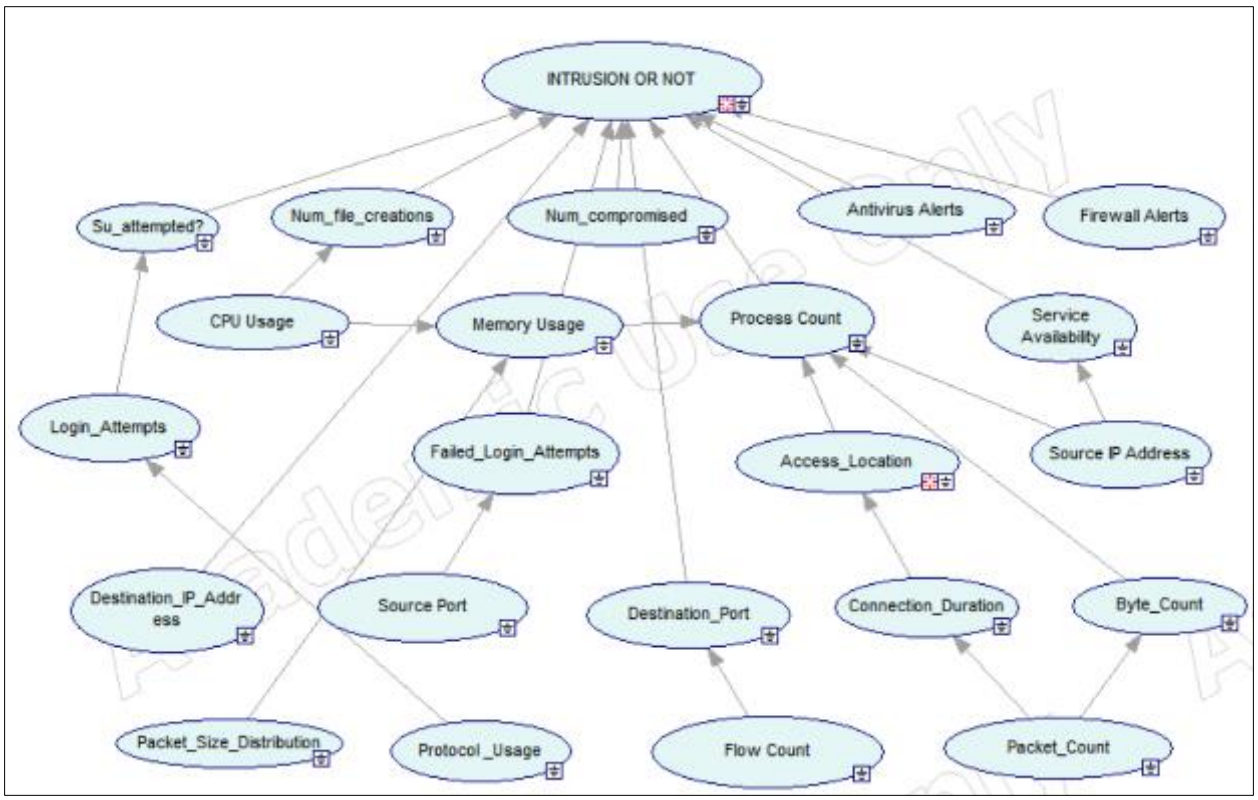

Figure 3 The Fully Quantified Bayesian model

The intrusion prediction model presented in this study was developed using the dataset detailed in Table 3.1. The network was created with GeNie Software, an open-source tool for implementing Bayesian networks, available at www.genie.sis.pitt.edu. The nodes in the network correspond to the features identified in Table 3.1.

Figure 3.3 illustrates the fully quantified Bayesian network model, where nodes represent selected factors from Table 3.1 and their associated conditional probability values. This structure is suitable for our research, as our primary focus is the status of the "INTRUSION OR NOT" node. GeNie Software was used to design this arrangement, which initially generates blank Conditional Probability Tables that were subsequently populated with the necessary conditional probabilities.

4.1. Influence and Scenario Analysis

The fully constructed and quantified Bayesian network model is then analyzed through Influence and Scenario analyses. These methods reveal how individual factors predict the target node. By populating the lower-level nodes with test data and setting the value of the target node (INTRUSION OR NOT) to 100%, influence analysis is performed to observe changes in the child nodes, as shown in Figure 3.4.

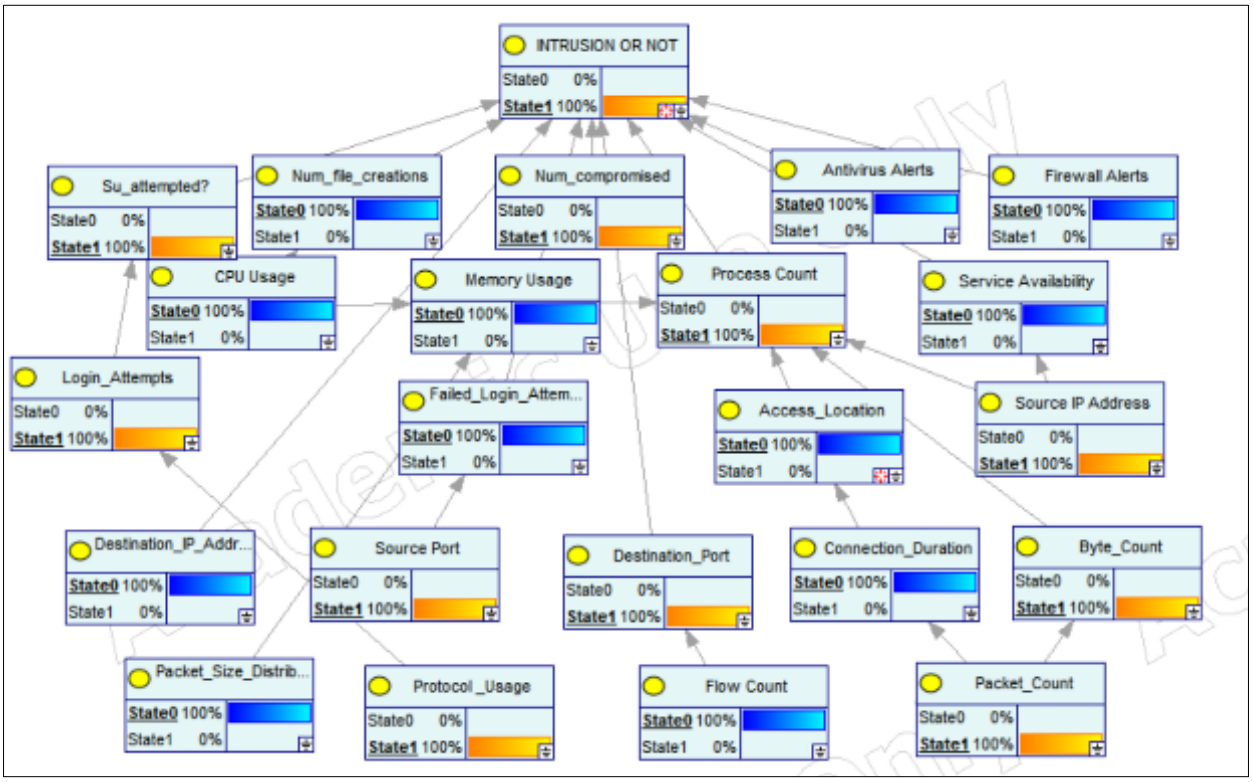

Figure 4 Influence and Scenario Analysis

5. Conclusion

This research work set out to build a predictive Bayesian Network model capable of detecting intrusion in a computer network. In order to achieve this task, the most influential features were selected from the widely used KDD dataset. These features formed the nodes of the bayesian network while their structural dependencies was established using the Max-Min Hill Climbing (MMHC) Algorithm. Scenario and influence analysis experimentation was carried out on the fully quantified model showed the effectiveness of the system using test data. While This work has focused on variable/features that predict occurrence of an intrusion and utilized data that was related to the network intrusion, with more research effort, this model could be generalized.

Compliance with ethical standards

Disclosure of conflict of interest

There was no conflict of interest.

References

[1] Acheme, I. D., & Enoyoze, E. (2024). Customer personality analysis and clustering for targeted marketing. International Journal of Science and Research Archive, 2024, 12(01), 3048–3057 https://doi.org/10.30574/ijsra.2024.12.1.1003.

[2] Acheme, I. D., & Vincent, O. R. (2021). Machine-learning models for predicting survivability in COVID-19 patients. In Data Science for COVID-19 (pp. 317-336). Academic Press.

[3] Acheme, I. D., Makinde, A. S., Udinn, O., & Nwankwo, W. (2020). An Intelligent Agent-Based Stock Market Decision Support System Using Fuzzy Logic. IUP Journal of Information Technology, 16(4).

[4] Acheme, I. D., Nwankwo, W., Olayinka, A. S., Makinde, A. S., & Nwankwo, C. P. (2023, March). Petroleum Drilling Monitoring and Optimization: Ranking the Rate of Penetration Using Machine Learning Algorithms. In The International Conference on Artificial Intelligence and Logistics Engineering (pp. 152-164). Cham: Springer Nature Switzerland.

[5] Acheme, I. D., Osemengbe, U., Makinde, A. S., & Vincent, O. R. (2021). Online Stores: Analysis of user Experience with Multiple Linear Regression Model. In 4th International Conference on Information Technology in Education and Development.

[6] Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1), e4150.

[7] Amouri, A., Alaparthy, V. T., & Morgera, S. D. (2020). A machine learning based intrusion detection system for mobile Internet of Things. *Sensors*, *20*(2), 461.

[8] Ijegwa, A. D., Olufunke, V. R., Folorunso, O., & Richard, J. B. (2019). A Bayesian based system for evaluating customer satisfaction in an online store. In *Intelligent Systems and Applications: Proceedings of the 2018 Intelligent Systems Conference (IntelliSys) Volume 2* (pp. 1047-1061). Springer International Publishing.

[9] Jaradat, A. S., Barhoush, M. M., & Easa, R. B. (2022). Network intrusion detection system: Machine learning approach. *Indones. J. Electr. Eng. Comput. Sci*, *25*(2), 1151.

[10] Kumar, S., Gupta, S., & Arora, S. (2021). Research trends in network-based intrusion detection systems: A review. *IEEE Access*, *9*, 157761-157779.

[11] MAKINDE, A. S., & ACHEME, I. D. (2023). Climate-Driven Maize Yield Prediction: A Machine Learning Approach.

[12] NSL-KDD Data set for Network-based Intrusion Detection Systems. Available at: http://nsl.cs.unb.ca/NSL-KDD.

[13] Saranya, T., Sridevi, S., Deisy, C., Chung, T. D., & Khan, M. A. (2020). Performance analysis of machine learning algorithms in intrusion detection system: A review. *Procedia Computer Science*, *171*, 1251-1260.

[14] Verma, A., Ranga, V. Machine Learning Based Intrusion Detection Systems for IoT Applications. *Wireless Pers Commun* 111, 2287–2310 (2020). https://doi.org/10.1007/s11277-019-06986-8