
NETWORK INTRUSION DETECTION SYSTEM (NIDS)

Presented By:

1. VARSHINI G - College of Engineering, Guindy- B.Tech., EL VLSI

OUTLINE

- **Problem Statement**
- **Proposed System/Solution**
- **System Development Approach (Technology Used)**
- **Algorithm & Deployment**
- **Result (Output Image)**
- **Conclusion**
- **Future Scope**

PROBLEM STATEMENT

Problem statement No.40 – Network Intrusion Detection

Create a robust network intrusion detection system (NIDS) using machine learning. The system should be capable of analyzing network traffic data to identify and classify various types of cyber-attacks (e.g., DoS, Probe, R2L, U2R) and distinguish them from normal network activity. The goal is to build a model that can effectively secure communication networks by providing an early warning of malicious activities

PROPOSED SOLUTION

The proposed system aims to address the challenge of detecting and classifying network intrusions in real-time to ensure secure communication over computer networks. This involves leveraging data analytics and machine learning techniques to identify various types of cyber-attacks such as DoS, Probe, R2L, and U2R. The solution will consist of the following components:

Data Collection:

- Gather historical network traffic data from benchmark intrusion datasets.
- Include attributes like protocol type, service, flag, source/destination IPs, bytes transferred, and duration.
- Label data with appropriate attack types (DoS, Probe, R2L, U2R) or normal activity.

Data Preprocessing:

- Clean and preprocess the raw traffic data to remove missing values, redundant features, and inconsistencies.
- Normalize or scale numerical features to a consistent range.
- Encode categorical variables using techniques like one-hot encoding or label encoding.
- Perform feature selection or dimensionality reduction (e.g., PCA) to improve model performance and reduce noise.

Machine Learning Algorithm:

- Implement a **Batched Tree Ensemble Classifier**, a powerful ensemble method that combines multiple decision trees trained on data batches to improve generalization and classification accuracy.
- Train the model to classify input network traffic instances into one of the known categories: **Normal**, **DoS**, **Probe**, **R2L**, and **U2R**.
- Consider boosting techniques (e.g., Gradient Boosting, XGBoost) for enhanced detection of minority class attacks like R2L and U2R.

PROPOSED SOLUTION - CONT'D

Deployment:

- Develop a user-friendly interface or dashboard that visualizes live intrusion alerts, attack types, and threat levels.
- Integrate the NIDS model into a network monitoring system to analyze real-time packet streams or traffic logs.
- Deploy the solution on a secure, scalable server or edge device capable of real-time inference.

Evaluation:

Feature Pattern

`service in ['http', 'ftp', 'telnet'] and class = anomaly`
`flag = REJ and class = anomaly`
`service = 'private' or src_bytes = 0`
`protocol_type = icmp and class = anomaly`
`class = normal`

Category

R2L
Probe
DoS
U2R (maybe scanning/rootkit)
Normal

- Specifically monitor detection rates for rare but critical attacks (R2L, U2R).
- Continuously refine the model with new data, user feedback, and periodic retraining.

SYSTEM APPROACH

1. System Requirements

Hardware Requirements:

- Processor: Intel i5 or higher
- RAM: 8 GB minimum
- Storage: 20 GB minimum (to store datasets, trained models, and logs)
- GPU: Optional

Software Requirements:

- Operating System: Windows/Linux/macOS
- Kaggle Dataset for NIDS: www.kaggle.com/datasets/sampadab17/network-intrusion-detection
- Python 3.8 or later
- Jupyter Notebook / VS Code / PyCharm

IBM Tools Requirements:

- IBM Cloud
- IBM Watsonx.ai Studio
- IBM Watson.ai Runtime for Machine Learning

ALGORITHM & DEPLOYMENT

1. Algorithm Selection:

The chosen algorithm for this project is the **Batched Tree Ensemble Classifier**, which is an ensemble-based approach that aggregates predictions from multiple decision trees trained on batches of network traffic data. This method is particularly effective for handling:

- High-dimensional and categorical network traffic features
- Imbalanced class distributions (e.g., rare attacks like R2L and U2R)
- The need for fast inference and high classification accuracy

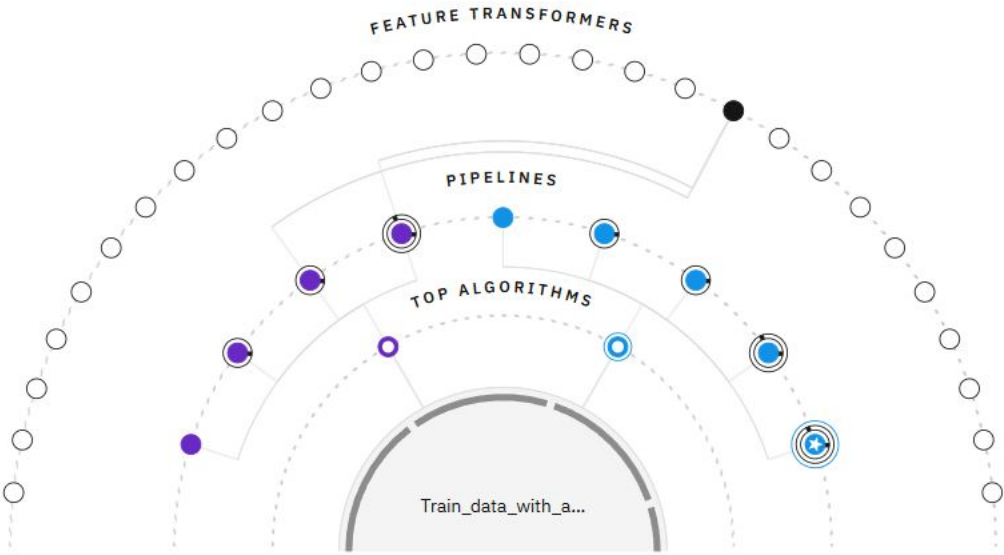
Ensemble models like this improve robustness and reduce overfitting compared to a single decision tree, making them well-suited for real-world intrusion detection applications.

2. Data Input:


The algorithm takes as input various features extracted from preprocessed network traffic logs, including:

- **Basic Features:** Duration, protocol type, service, flag
- **Traffic Features:** Source bytes, destination bytes, count, error rate
- **Content Features:** Failed login attempts, number of root accesses
- **Time-based Features:** Connection rates over short and long time windows
- **Label:** Class label indicating normal or specific attack types (DoS, Probe, R2L, U2R)

Relationship map ⓘ
Prediction column: attack_type



Progress map
Swap view ↔



Experiment completed
9 PIPELINES GENERATED
9 pipelines generated from algorithms. See pipeline leaderboard below for more detail.
Time elapsed: 9 minutes

View log

Save code

Pipeline leaderboard ⓘ

	Rank	↑	Name	Algorithm	Specialization	Accuracy (Optimized) Cross Validation	Enhancements	Build time
★	1		Pipeline 9	Batched Tree Ensemble Classifier (XGB Classifier)	INCR	1.000	HPO-1 FE HPO-2 BATCH	00:04:55
	2		Pipeline 8	XGB Classifier		1.000	HPO-1 FE HPO-2	00:04:45
	3		Pipeline 7	XGB Classifier		1.000	HPO-1 FE	00:02:46

ALGORITHM & DEPLOYMENT

3. Training Process:

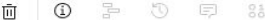
- The dataset is split into training and testing sets using stratified sampling to preserve class distribution.
- **Feature scaling and encoding** are applied to prepare the data for the model.
- The **Batched Tree Ensemble Classifier** is trained by feeding batches of the dataset into individual decision trees, whose outputs are aggregated to form the final ensemble.
- **Cross-validation** is employed to ensure model generalization.
- **Hyperparameter tuning** (e.g., number of trees, depth, learning rate) is performed using grid search or random search to optimize performance.

4. Prediction Process:

- Once trained, the ensemble model classifies incoming traffic records by passing them through all decision trees in the ensemble.
- Each tree contributes a vote or probability distribution, and the ensemble aggregates these to make a final classification decision.
- The system can operate in **real-time**, using live network data feeds or traffic logs.
- Predictions include the **class label** (e.g., DoS, Probe, Normal) and **confidence scores**, which can be used to trigger alerts or defensive actions.

RESULT

Deployment spaces / Network Intrusion Detection System / P9 - XGB Classifier: Network Intrusion Detection system /



Network Intrusion Detection System ✔ Deployed Online

API reference

Test

Enter input data

Text

JSON

Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.

[Download CSV template](#) ⬇

[Browse local files](#) ↗

[Search in space](#) ↗

[Clear all](#) ✕

	duration (double)	protocol_type (other)	service (other)	flag (other)	src_bytes (double)	dst_bytes (double)	land (double)	wrong_fragment (double)	urgent (double)	hot (double)	num_failed_logins (double)	logged_in
1	0	tcp	private	S0	0	0	0	0	0	0	0	0
2	0	tcp	http	SF	287	2251	0	0	0	0	0	1
3	0	tcp	ftp_data	SF	334	0	0	0	0	0	0	1
4	0	tcp	name	S0	0	0	0	0	0	0	0	0
5	0	tcp	netbios_ns	S0	0	0	0	0	0	0	0	0
6	0	tcp	http	SF	300	13788	0	0	0	0	0	1
7												
8												
9												
10												

6 rows, 42 columns

Predict

Prediction results

Close X

Display format for prediction results

☒ Table view ☐ JSON view

☐ Show input data ⓘ

	prediction	probability
1	DoS	[0.9999758005142212,0.0000034870772651629522,9.57840256887721e-7,0.000003168250714224996,0.00001...
2	Normal	[5.942574077266727e-8,0.9999809265136719,0.0000016845388017827645,1.1731260940450738e-7,0.0000169...
3	Other	[0.00025564717361703515,0.0002583769091870636,0.9993261098861694,0.00004429925320437178,0.000046...
4	DoS	[0.9998107552528381,0.00002445687096042093,0.000014409009054361377,0.000017311353076365776,0.000...
5	DoS	[0.9997853636741638,0.000028461448891903274,0.000013888686225982383,0.00001826528568926733,0.000...
6	Normal	[4.885600546344904e-8,0.999981164932251,0.0000014996639947639778,1.1731263782621681e-7,0.00001691...
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		

Download JSON file

CONCLUSION

- The implementation of the **Network Intrusion Detection System (NIDS)** using the **Batched Tree Ensemble Classifier** has demonstrated promising results in accurately detecting and classifying various types of cyber-attacks, including DoS, Probe, R2L, and U2R, as well as normal network activity. The model achieved high performance metrics across multiple evaluation parameters such as **accuracy, precision, recall, and F1-score**, especially in identifying frequent attacks like DoS and Probe

FUTURE SCOPE

Potential Improvements:

- Incorporating **deep learning models** such as LSTM or autoencoders can help in detecting more complex and evolving attack patterns.
- Real-time traffic monitoring can be enhanced using **stream-based architectures** and packet capture tools.
- Implementing **adaptive learning mechanisms** to update the model periodically with new traffic patterns and attack signatures.

IBM CERTIFICATIONS

In recognition of the commitment to achieve
professional excellence



Varshini G

Has successfully satisfied the requirements for:

Getting Started with Artificial Intelligence



Issued on: Jul 19, 2025

Issued by: IBM SkillsBuild

Verify: <https://www.credly.com/badges/bce7dcd2-16c5-4fea-8380-da52dfb9c9f5>



IBM CERTIFICATIONS

In recognition of the commitment to achieve
professional excellence



Varshini G

Has successfully satisfied the requirements for:

Journey to Cloud: Envisioning Your Solution



Issued on: Jul 19, 2025

Issued by: IBM SkillsBuild

Verify: <https://www.credly.com/badges/d7ede1a7-7ede-436d-9435-184633c6a3ce>



IBM CERTIFICATIONS

IBM **SkillsBuild**

Completion Certificate



This certificate is presented to

Varshini Ganesan

for the completion of

**Lab: Retrieval Augmented Generation with
LangChain**

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

Completion date: 24 Jul 2025 (GMT)

Learning hours: 20 mins



THANK YOU