TEST PLAN FOR

NETWORK INTRUSION DETECTION SYSTEM

Made by:
Shikhar Raj 2000290120144
Shubhi 2000290120160
Abhijeet kannujia 2000290120007

Test Manager:
Prof. Shreela Pareek

Project Co-Ordinator
Prof. Sreesh Gaur

Submitted to:
Prof. Neha Shukla

ChangeLog

Version	Change Date	By	Description
version number	Date of Change	Name of person who made changes	Description of the changes made
1.0	30-9-23	Shikhar Raj	Initial draft
1.1	25-10-23	Shubhi	Added login functionality

1 Introduction

In the todays digital era highly of sophisticated technology , the amount of cyber-attacks has comprised the user's data and security.
The Network intruison detection system is an application which is being used to detect the intruisons and classify the attack class (normal , DOS ,Probe)
Test Strategies:
The testing phase of the project aims to ensure the robustness, accuracy, and generalization of the developed model for the attack classification in network intrusion. The test strategies involve a combination of quantitative metrics, qualitative assessments, and validation against ground truth data.

Process:

1.Data Preprocessing
-Acquired KDD Cup 1999 data set is a database of data to be audited that includes a variety of intrusions simulated in a military network environment.
-Preprocessed the data, including normalization, to enhance model generalization.
2.Model Training
-The KDD CUP 1999 dataset into a training and testing set of 65:35, and then the Random Forest Classifier of ensemble learning to train the model.
-The model is saved by using a pickle, which we further used to predict the packets in testing
-Split the dataset into training and validation sets to monitor model performance during training.
3.Model Evaluation
-After training , the model is evaluated on the testing data to assess its performance. The accuracy_score function is used to calculate the accuracy of the model's predictions compared to the actual values.
4.Validation
-Validated the trained model on separate test data not used during training to assess its ability to generalize to unseen data .

Workflow:

1.Input Data

-Received various network parameters for classification , ensuring diversity and representation of different label type.

2.Data Preprocessing

-Applied preprocessing techniques to handle variations in the attack classes.

3.Model Architecture

-Ensemble methods combine the outputs of multiple individual models to improve detection accuracy. Techniques like Random Forests, Gradient Boosting, or stacking can be used to create ensemble models that classify network traffic into attack classes.

4.Training

-Divided the dataset into training, and test sets.

-Trained the model using the training set, testing on the test cases.

5.Evaluation

- Evaluated the trained model on the test set to assess its performance in actual scenarios

6.PostProcessing

-may involve generating reports and documentation summarizing the alerts and incidents detected.

-Applied post-processing techniques to refine the output masks and enhance the final classification results

Methodologies

1.Data Preparation and Labeling:

-Gather and preprocess network traffic data, extract relevant features, and label data based on known attack classes and "normal" traffic.

2.Model Selection and Training:

-Choose appropriate detection models or algorithms, train machine learning models, and define attack signatures or rules for classification.

3.Real-time Detection and Alerting:

-Deploy the NIDS to monitor network traffic, classify traffic into attack classes, and generate alerts when attacks are detected.

1.1 Scope

1.1.1 In Scope

Scope defines the features, functional or non-functional requirements of the software that will be tested. Certainly! Defining the scope of testing for your web-based project involves outlining the features, functional and non-functional requirements, and specific aspects of the software that will be subjected to testing. Below is a breakdown of how you might define the scope for testing:

Features:

1.User Interface (UI):

- Testing the responsiveness of the web application across different devices and screen sizes

- Checking of the loading time and performance of the Interface in different hardware specifications.

2. Navigation :

- Confirm that all the modules are working effectively in a systematic manner.

- Check the accessibility and usability of the navigation menu.

3. Functionality :

- Validate the functionality of the critical features implemented such as login authentication , data processing and the attack class prediction.

- Ensure that user inputs(Testcases) are correctly processed and validated

Functional Requirements

1.Compatibility Testing

- Test the web application on different browser such as chrome firefox safari to ensure cross- browser compatibility

- check the working of the project on different operating systems

2. Error Handling

- Verify that appropriate error message is displayed for invalid inputs

- Testing the application under unexpected scenarios.

3. Blackbox testing

- In this approach, simulate various attacks without detailed knowledge of the NIDS's internal workings, focusing on how it responds to external threats.

Non – Functional Requirements

1. Usability:

- Conduct usability testing to assess the user-friendliness of the application.

- Check the clarity of instructions and the intuitiveness of workflows.

2. Performance:

- Validate that the web application meets specified performance benchmarks.

- Check resource usage and response times under varying conditions

3 . Scalability Testing

-In-depth testing to assess the system's scalability and ability to handle a growing user base is not within the immediate scope. This may be considered in future testing phases

1.1.2 Out of Scope

1. Browser Compatibility Beyond Specification:
- Testing on browsers not specified in the compatibility matrix.
2. Obsolete Operating Systems:
- Testing on operating systems that are no longer supported or widely used.
3. Network Speeds and Conditions:
- Extensive testing under extreme network conditions or speeds beyond the defined criteria.
4. Hardware Variations:
- Testing on hardware configurations not specified in the project requirements.
5. Non-Standard Devices:
- Testing on devices not within the standard scope, such as smart TVs, gaming consoles, etc., unless explicitly stated.

1.2 Quality Objective

Adherence to Project Timeline:
Ensure that the development and deployment of the application adhere to the agreed-upon project timeline. Timely delivery is crucial for project success and stakeholder satisfaction.
Scalability and Future-Proofing:
Design the application with scalability in mind to accommodate future growth and changes in user requirements. This objective ensures that the software remains relevant and adaptable over time.
Continuous Improvement:
Foster a culture of continuous improvement within the development and deployment processes. Learning from each project iteration ensures ongoing enhancement of development practices.

1.3 Roles and Responsibilities

Detail description of the Roles and responsibilities of different team members like

- Test Manager – Prof. Shreela Pareek
- Project Manager – Prof. Sreesh Gaur
- Configuration Manager – Prof. Neha Shukla
- Developers - Shikhar Raj, Shubhi, Abhijeet kannaujia
- Installation Team- Shikhar Raj, Shubhi, Abhijeet kannaujia Prof. Shreela Pareek , Prof. Neha Shukla
- QA Analyst – Shikhar Raj, Shubhi, Abhijeet kannaujia

2 Test Methodology

2.1 Overview

Reasons for Adopting Waterfall

Structured Phases:

The Waterfall model follows a clear and sequential structure, with defined phases for requirements, design, implementation, testing, deployment, and maintenance. This structured approach aligns well with projects that have well-defined and stable requirements.

Thorough Planning Emphasis:

Waterfall places a strong emphasis on meticulous planning and extensive documentation in the initial stages. This proves critical for your project, where a comprehensive understanding of vital components such as user authentication, and location tracking features is imperative.

Detailed Requirements Management:

The Waterfall model excels when requirements are firmly established and less likely to undergo significant alterations throughout the project lifecycle. For functionalities like the intrusion tracking in our case, detailed specifications that undergo infrequent revisions are pivotal.

Systematic Testing Approach:

Waterfall's phased approach allows for structured testing after the development phase concludes. This is particularly beneficial for your project, ensuring that individual components like user authentication undergo rigorous testing before progressing to subsequent phases (operational network intrusion system).

2.2 Test Levels

Following are the testing levels that are defined based on the case of the threat detection and neutralisation System:

Requirements Testing: Ensuring that the system's requirements, including security and regulatory aspects, are met.

Design Testing: Verifying that the system's design adheres to the specifications and is suitable for attack classification.

Implementation Testing: Testing the developed system to ensure it meets the design specifications, is secure, and functions as intended.

System Testing: Evaluating the entire system to verify its correctness and compliance with election regulations.

User Acceptance Testing (UAT): Allowing end-users, to validate the system for usability and suitability

2.3 Bug Triage

Project Name – threat detection system
Version – 1.1
Date : 25/10/23
Reported by – Shikhar raj
BUG Details
Severity : medium
Priority : medium
Status :closed
Description : Alphabetical values
During the execution of the test case 9 and test case 10 , which is aimed at validating the threat detection functionality based on predefined criteria , it was observed that the system went on the internal server error resulting in termination of the application

2.4 Suspension Criteria and Resumption Requirements

1. Suspension Criteria
-Threshold-based Suspicion: Set specific thresholds for various network parameters (e.g., packet rate, connection rate, bandwidth usage). When any of these thresholds are exceeded, the NIDS can flag the activity as suspicious.
-Anomaly Detection: Employ machine learning or statistical analysis to create a baseline of "normal" network behavior. When the system detects significant deviations from this baseline, it can trigger an alert.
-Signature Matching: Use pre-defined attack signatures or patterns to identify known attacks. If incoming traffic matches these signatures, it can be flagged as an intrusion attempt.
-Behavioral Analysis: Analyze the behavior of network traffic, such as port scanning or repeated login failures. If certain patterns of behavior are detected, it can be a reason for suspicion.

2.Resumption Requirements:
- Resumption requirements specify the conditions and procedures that must be met for testing to resume after it has been suspended. These requirements ensure that testing can continue effectively once the suspension criteria have been addressed. For threat detection system the resumption requirements are as follows:
-Event Logging**:** Record detailed information about the detected attack or intrusion attempt, including timestamps, source and destination IP addresses, the attack type, and any relevant payload data.
- Continuous Monitoring: Resume continuous monitoring of the network to detect and prevent future intrusion attempts. This includes keeping an eye on security alerts and adjusting the NIDS as needed.
-Documentation: Document all actions taken during the resumption process. This documentation is crucial for post-incident analysis, compliance requirements, and potential legal or regulatory purposes.

2.5 Test Completeness

- Here you define the criteria that will deem your testing complete.
- For instance, a few criteria to check Test Completeness would be
- 100% test coverage
- All Manual & Automated Test cases executed.
- All open bugs are fixed or will be fixed in next release.

3 Test Deliverables

- In This phase the test cases are generated based upon the manual testing in which the input is being processed by the tester
- The boundary value analysis is being done based on the formula 4N+1 where N is the variable. The value of N is taken to be 3 because there are 3 attack classes (normal, DOS, probe) and these are being predicted and they are the variables.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	ID	count	dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_same_srv_rate	dst_host_srv_count	Status of connection	Last flag	logged in	same _srv_rate	serror rate	http used or not	attack	attack predicted
2	1	2	0.6	0	0	0.5	S0	1	0	0.7	0.15	yes	satan	probe
3	2	1	0.05098	0.2766	0.13	0.1395	SF	0.14	1	0.085	0.875	yes	satan	probe
4	3	0	0	0	0	0	S0	0	0	0	0	no	normal	normal
5	4	0	0	0	0	0	SF	0	0	0	0	yes	normal	normal
6	5	101	0.1	0.91	0.96	106	SF	1	0	0.91	0.04	no	satan	probe
7	6	255	0.01	0.91	0.96	155	S0	1	0	0.97	0.03	no	neptune	DOS
8	7	2	0.98	0.05	0.02	1	SF	1	0	0.03	0.1	no	satan	probe
9	8	1	0.95	0.05	0.02	1	S0	0	0	0.03	0.1	no	neptune	probe
10	9	a	b	c	d	e	SF	1	1	b	c	yes	normal	Error
11	10	1a	3d	7c	e	fg	SF	0	0	0	1	no	neptune	Error
12	11	225	0.06	0.05	0.08	0	S0	0	1	1	0.1	no	neptune	DOS
13	12	1	1	1	1	1	S0	1	1	1	0	NO	normal	normal
14	13	10	0.9	0.1	0.92	11	SF	1	0	0	0.05	no	neptune	DOS
15														
16														
17														

1. Decision table for User Login

	INPUT1	INPUT2	INPUT3	INPUT4
Email	T	T	F	F
Password	T	F	T	F
Login(Y/N)	Y	N	N	N

2. Equivalence class Partioning for the network intrusion detection system

Packet Size	HTTP Used	Attack class
Large >1000 bytes	NO	DOS
Small<(1000 bytes)	YES	Normal
0 bytes < Range<1000bytes	YES	Probe

4 Resource & Environment Needs

4.1 Test Environment

It mentions the minimum hardware requirements that will be used to test the Application.
Processor: Intel core i5 /Ryzen 5 or equivalent
RAM: 8GB minimum
Storage: 256 Gb SSD or 1tb hard disk drive
Graphic Card: Nvidia GeForce / AMD Radeon 4gb VRAM
Display: Minimum of 1920*1080p
Internet connectivity required.

SOFTWARE Requirements

The minimum software requirements necessary for perfect working
1.Operating system – windows 10 (64bit)
2.Web Browser – (google chrome, Firefox, safari)
3.Python – version 3.6
4.Visual studio code
5.Jupyter Notebook (ML models)
Testing Framework – Selenium
Note: The specified software versions are the minimum requirements. It is advisable to use the latest stable releases of the mentioned software for optimal performance and compatibility.

5 Terms/Acronyms

Make a mention of any terms or acronyms used in the project

TERM/ACRONYM	DEFINITION
API	Application Program Interface
AUT	Application Under Test