ARTIFICIAL INTELLIGENCE /MACHINE LEARNING FOR NETWORK SECURITY

Submitted By

Hafeez- (230051601049)

B.S Afroz Mohammed - (230051601006)

Anbu - (230051601009)

INTRODUCTION

Artificial Intelligence (AI) and Machine Learning (ML) have a major impact on almost every organization and individual. Things changed quickly with the rise of generative AI and when OpenAI released ChatGPT to the general public in November 2022. In the AI and ML lesson, we discussed the differences between AI and ML and the different AI types.

AI and ML are also used in networking. Networking complexity has increased because we have larger networks, with multi-tenancy and more virtualization, such as overlay networks. Modern networks require more than traditional rule-based management systems. AI and ML allow us to analyze data in real time, identify patterns, predict issues, and even automate decision-making.

PROBLEM STATEMENT

"How can we build an efficient and accurate intrusion detection system that identifies SQLi and XSS attacks from web traffic or URLs in real time, using AI/ML models for feature extraction and classification — with minimal latency and high scalability?"

Core Problem

Modern networks are vulnerable to attacks like SQL Injection (SQLi) and Cross-Site Scripting (XSS) due to the increasing complexity of traffic and widespread use of web applications.

How SQLi and XSS Attacks Affect Us (as individuals and organizations)

From the perspective of both end-users and organizations, SQL Injection (SQLi) and Cross-Site Scripting (XSS) are highly dangerous. The webinar PDF highlights the payload risks, and here's how they affect us in practical, real-world ways

SQLi

An attacker manipulates database queries by injecting malicious SQL code — e.g., using a login form to bypass authentication (' OR 1=1 --).

Impact:

1. Data Theft

2. Loss of Data Integrity

3. Authentication Bypass

4. Reputation Damage & Legal Fines

XSS

Attackers inject malicious JavaScript into webpages that other users visit, often via forms, URLs, or comments.

Impact:

1. Session Hijacking

2. Credential Theft & Phishing

3. Defacement of Websites

4. Spreading Malware

Attack	Security Impact	Business Impact
SQLi	Database Compromise	Data Breach, Legal Consequences
XSS	Browser Hijack, UI Manipulation	Customer Trust Loss, Financial Damage

Problem Solution: SQLi and XSS Detection Using AI/ML

To solve the problem of detecting SQL Injection (SQLi) and Cross-Site Scripting (XSS) attacks, we followed a modern, AI-driven approach as explained in the Unnati Ignite Webinar and implemented in Streamlit app.

Step-by-Step Solution Using AI/ML

1. Problem Identification

- Challenge: Traditional rule-based systems (regex, signatures) fail to detect evolving SQLi/XSS attacks.
- Goal: Automatically and accurately identify malicious URLs and traffic patterns.

2. Feature Extraction

We extract useful features from URLs or HTTP payloads to feed into machine learning models.

Extracted Feature Types: | Feature Type | Example | Purpose | |--------------|---------|---------| | **Lexical** | URL length, special characters (=, ', <script>) | Detect SQLi/XSS syntax | | **Statistical** | Token count, entropy | Catch patterns common in obfuscated attacks | | **Payload-based** | HTTP Host, SNI, DNS | Understand deeper request context |

3. Model Training & Selection

We train multiple AI models to classify URLs as **malicious (1)** or **benign (0)**.

Model	Accuracy (from PDF)	Notes
Deep Learning (URLNet)	99.78%	Highest accuracy but slower
Random Forest	96.21%	Excellent balance of speed and accuracy
Libinjection	97.17%	Ruleset-based; reliable but not adaptive
Regex	83.23%	Fast but outdated

4. Inference & Detection

In the app:

- When a user submits a URL:
  - extract_features() converts the URL into a feature vector.
  - The Random Forest model predicts 0 (safe) or 1 (malicious).
  - Display result to the user: ✅ Safe or ❌ Unsafe (SQLi/XSS risk).

5. Real-Time Feedback (in Streamlit App)

You provide instant feedback for:

- **Single URL** (input box)
- **Batch URLs** (CSV upload)

This makes your app function like a lightweight AI-enabled Web Application Firewall (WAF).

6. Optional Enhancements

You can further extend detection using:

- Deep learning models (e.g., URLNet or R1DIT-CNN via OpenVINO/TensorFlow)
- VPN/malware traffic classification
- App identification (Skype, YouTube) using statistical or entropy-based pattern.

Step	Action	Impact
1. Understand threat	Focus on SQLi/XSS	Clear scope
2. Extract features	Token-based, lexical, entropy	Learn attack patterns
3. Train model	Random Forest	High accuracy (96%+)
4. Predict URL safety	In real time	Stop attacks before they spread
5. Show results	Clear user messages	Usable interface

Algorithm of the code

Input:

- User selects input method:
  - A single URL (manual input)
  - A CSV file with multiple URLs
- Pre-trained model (rf_model.pkl)
- Feature extractor function (extract_features())

Output:

- Prediction result for each URL:
  - "Your Link is Secure"
  - "Unsafe URL Found"

Step-by-Step Algorithm

Step 1: Load Model

1. Load the trained Random Forest model using joblib.

model = joblib.load("model/rf_model.pkl")

Step 2: Display Title and Input Option

1. Display the app title.
2. Ask the user to choose input method: "Upload URLs" or "Enter Single URL".

Step 3: Handle Input Based on Mode

If Upload URLs mode:

1. Let the user upload a CSV file.
2. Read the CSV and extract the url column.
3. For each URL:
   - Extract features using extract_features().
   - Use the model to predict the label (0 = safe, 1 = malicious).

1. Map predictions to readable labels:
   - 0 → ✅ Your Link is Secure
   - 1 → 🚫 Unsafe URL Found

1. Display the results in a table.
2. If Enter Single URL mode:
3. Wait for the user to type in a URL.
4. If URL is entered:
   - Extract features using extract_features().
   - Predict using the model.

1. If the result is:

- 1 → Show red warning box with "Malicious Link Detected"
- 0 → Show green box with "Your Link is Secure"

Code we used

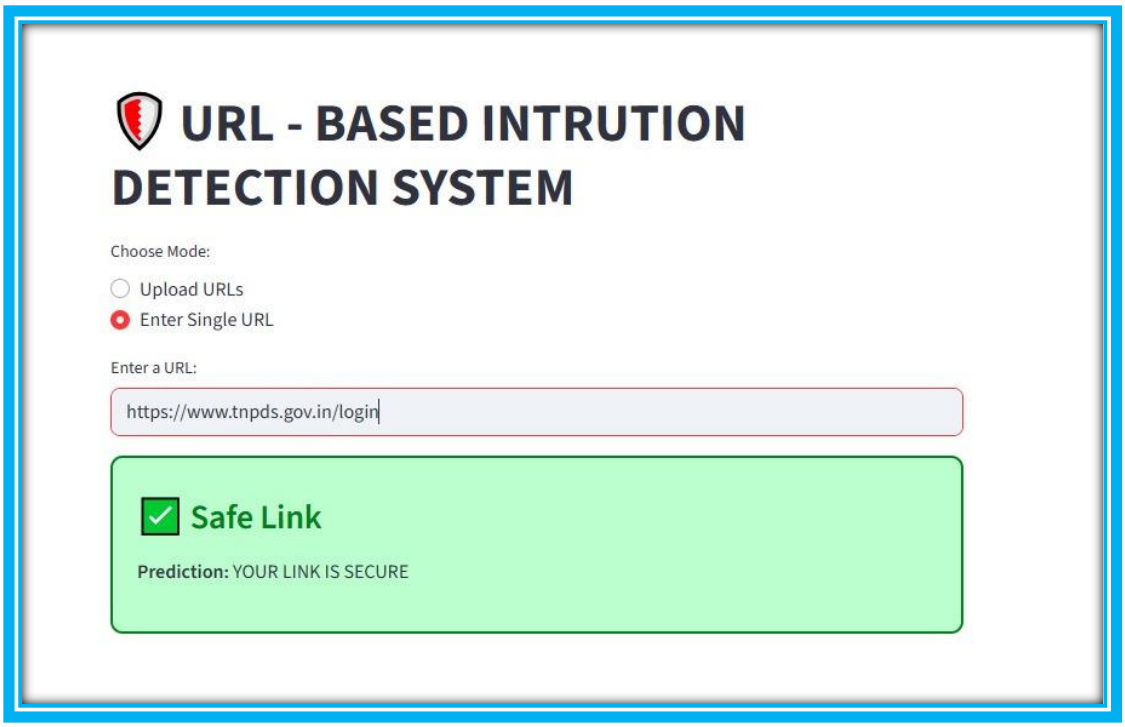
```
import streamlit as st
import pandas as pd
import joblib
from utils.feature_extraction import extract_features
st.title("🛡️ URL - BASED INTRUTION DETECTION SYSTEM ")
model = joblib.load("model/rf_model.pkl")
option = st.radio("Choose Mode:", ("Upload URLs", "Enter Single URL"))
if option == "Upload URLs":
    uploaded_file = st.file_uploader("Upload CSV with column 'url'", type=["csv"])
    if uploaded_file:
        df = pd.read_csv(uploaded_file)
        df["features"] = df["url"].apply(extract_features)
        df["prediction"] = df["features"].apply(lambda x: model.predict([x])[0])
        df["result"] = df["prediction"].map({1: "🚫 UNSAFE URL FOUND", 0: "✅ YOUR LINK IS SECURE"})
        st.dataframe(df[["url", "result"]])
else:
    url = st.text_input("Enter a URL:")
    if url:
        features = extract_features(url)
        pred = model.predict([features])[0]
     if pred:
            # Malicious case
            st.markdown(
                f"""
                <div style='background-color:#ffcccc; padding:20px; border-radius:10px; border:2px solid red;'>
                    <h3 style='color:red;'>⚠️ Warning: Malicious Link Detected</h3>
                    <p><strong>Prediction:</strong> 🔴 UNSAFE URL FOUND</p>
                </div>
                """,
                unsafe_allow_html=True
            )
        else:
            # Safe case
            st.markdown(
                f"""
                <div style='background-color:#ccffcc; padding:20px; border-radius:10px; border:2px solid green;'>
                    <h3 style='color:green;'>✅ Safe Link</h3>
                    <p><strong>Prediction:</strong> YOUR LINK IS SECURE</p>
```

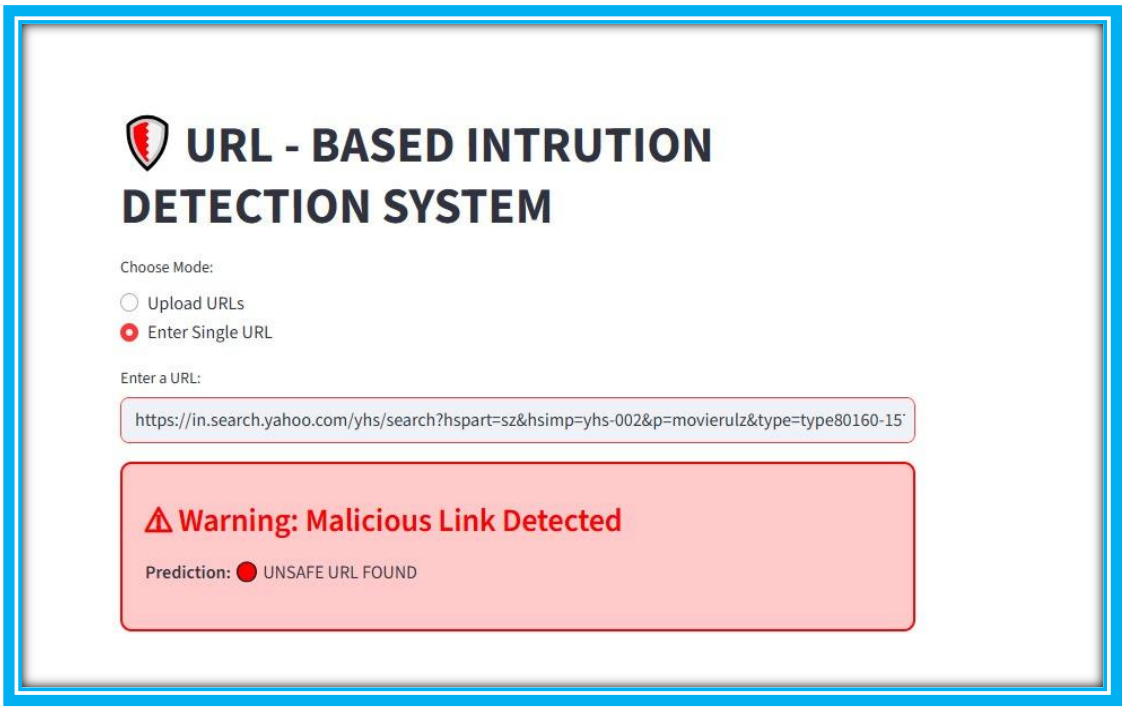
```
        </div>
        """,
        unsafe_allow_html=True
    )
```

How the code works

- loads your pre-trained Random Forest model.
- Lets the user choose between single or multiple URL input.
- This extracts features from each URL and
- Then uses your model to predict if it's malicious
- Shows a red warning box if the model thinks the URL is unsafe.

Output





Source and links

- ***http://192.168.12.208:8501***

Git hub

- https://github.com/hafeez916/ai-network-security.git