# Real-Time Breach Alerts using Machine Learning

## Problem It Solves and Its Relevance

Modern computer networks are constantly exposed to cyber threats such as unauthorized access, data breaches, and DoS attacks. Manually monitoring such threats is time-consuming and often ineffective. This project uses machine learning to build a Real-Time Breach Alert System that can automatically detect malicious activity and notify users immediately.

The system is built using the UNSW-NB15 dataset, which includes detailed records of network activity labeled as either normal or malicious. By training a machine learning model to recognize the patterns of these records, the solution helps users detect potential security breaches in real-time.

## Dataset Overview

Dataset: UNSW-NB15
Source: Australian Centre for Cyber Security
Records: ~2.5 million
Features per record: 100+
Types of traffic: Normal + 9 attack categories (e.g., DoS, Shellcode, Worms)

## Key features used:

- dur: Duration of connection
- sbytes, dbytes: Byte size in each direction
- rate, sload, dload: Data rates and loads
- sttl, dttl: Time-to-live values
- tcprtt, synack, ackdat: TCP-specific handshake metrics
- proto, service, state: Protocol and connection status

## Target variable:

- label: Binary (0 = Normal, 1 = Attack)

## Model Selection and Performance

Initial Attempt – Isolation Forest (Unsupervised)

We first tried using an Isolation Forest, which is good for anomaly detection. However, performance was poor:

- Accuracy: 38%
- Recall (Attack): 7%

The model missed most attacks because it could not learn from labeled data.

- Final Model – Random Forest (Supervised)

We switched to a Random Forest classifier, a supervised method, and balanced the dataset using SMOTE. This resulted in a major improvement:

Performance Metrics:

- Accuracy: 94%
- Precision (Attack): 96%
- Recall (Attack): 92%
- F1-Score (Attack): 94%

Confusion Matrix:

```
                 Predicted
               Normal  Attack
Actual Normal   31848    1129
Actual Attack    2571   30322
```

The model now reliably identifies attacks and minimizes false positives.

## Integration with the Prototype

The machine learning model is integrated into a user-focused security application. Here's how it works:

1. Input: The app continuously monitors user network activity (e.g., packets, logins).
2. ML Processing: Each connection is passed through the trained Random Forest model.
3. Detection:
   - If normal → no action.
   - If anomalous → trigger breach alert.
4. Alert System: Real-time notifications (e.g., push or dashboard alert) are sent to the user.

This system ensures proactive security and helps users act quickly before real damage occurs.

## How the Model Uses the Selected Features

During Training :

The Random Forest model is an ensemble of decision trees. Each tree learns simple rules using feature thresholds to classify traffic as normal or attack.

Example rule learned:

```
if proto == 'tcp' AND sbytes > 5000 AND rate > 0.8:
    → Predict: Attack
else:
    → Predict: Normal
```

- Trees vote, and the majority decision is used.
- It learns interactions like: proto = udp + sload high = suspicious.

Real-Time Prediction Flow

1. Feature Extraction:
   - Collect dur, proto, sbytes, rate, etc. from a new connection.
2. Preprocessing:
   - Encode categorical values, scale numeric values.
3. Prediction:
   - Pass values through the trained model.
   - Model votes based on its learned patterns.
4. Decision:
   - Predicts 1 (attack) or 0 (normal).
   - Triggers alert if it's an attack.

Real Example Patterns Learned:

| Situation | Feature Behavior | Model Decision |
|---|---|---|
| Data exfiltration | sbytes and sload very high | Attack |
| Port scanning | dur very low, rate high, sttl low | Attack |
| Normal web traffic | proto = TCP, balanced flow, low rate | |
| Flood attack | | |

| Situation | Feature Behavior | Model Decision |
|---|---|---|
| Data exfiltration | sbytes and sload very high | 🚨 Attack |
| Port scanning | dur very low, rate high, sttl low | 🚨 Attack |

| Situation | Feature Behavior | Model Decision |
|---|---|---|
| Normal web traffic | proto = TCP, balanced flow, low rate | ✅ Normal |
| Flood attack | rate very high, dbytes = 0, state = CON | 🚨 Attack |

✅ Why Random Forest Works Well

- Learns from labeled data
- Handles categorical and numeric features
- Captures non-linear relationships
- Resistant to noise and overfitting

🕹️ Summary

- We solved a real-world problem: detecting breaches in real-time.
- We tested Isolation Forest first – performance was poor.
- Then we used Random Forest with SMOTE – performance was excellent.
- The model is now integrated into a working prototype to protect users in real-time.

This project shows how machine learning can enhance cybersecurity by providing intelligent and timely threat detection.