This project is an **Intrusion Detection System (IDS)** that:

1. **Trains a Machine Learning model** (Random Forest) using the **NSL-KDD dataset** (a benchmark dataset for network intrusion detection).
2. **Extracts features from live network packets** (using **Scapy**).
3. **Classifies packets** as either **normal** or **attack** using the trained ML model.
4. **Logs intrusions** when suspicious traffic is detected.

Basically, it watches your network traffic in real time and tries to detect hacking attempts.

This is a **basic Intrusion Detection System (IDS)** that combines **ML (Random Forest)** + **packet sniffing (Scapy)** to detect suspicious packets in real time.

File by File Explanation

**train\_model.py**

* **Purpose**: Train the ML model using the NSL-KDD dataset.
* **Steps**:
  1. Loads **KDDTrain+ dataset** (a labeled dataset of network traffic).
  2. Uses only **5 features** (protocol\_type, service, flag, src\_bytes, dst\_bytes).
  3. Encodes categorical values (tcp, http, SF etc.) into numbers with **LabelEncoder**.
  4. Splits data into training & testing sets.
  5. Trains a **Random Forest Classifier**.
  6. Evaluates accuracy using a classification report.
  7. Saves the **trained model + encoders** as ids\_model\_simple.pkl.

After running this file once, you have a model ready to detect intrusions.

**feature\_extraction.py**

* **Purpose**: Extract numerical features from **raw packets** captured by Scapy.
* **How it works**:
  + If the packet is **IP-based**:
    - Reads source/destination IP, protocol number, packet length.
    - If TCP/UDP/ICMP → extracts port numbers & flags.
  + Returns a **list of features** [proto, len, sport, dport, flags].

This file isn’t directly used in detection. It seems like a utility or earlier version.

**detect\_intrusion.py**

* **Purpose**: Use the trained model to detect if a packet is an **attack**.
* **How it works**:
  1. Loads the trained model (ids\_model\_simple.pkl).
  2. **Extracts features** from the packet (protocol, service, flag, src\_bytes, dst\_bytes).
  3. Uses the saved encoders to convert them into numerical values.
  4. Passes features into the model (clf.predict).
  5. If classified as "attack":
     + Logs it into logs/intrusion\_logs.txt.

So this is the **core detection logic**.

**capture.py**

* **Purpose**: Sniff live packets and send them to the detector.
* **How it works**:
  + Uses Scapy’s sniff() function to capture packets.
  + Every time a packet arrives:
    - If it’s TCP/UDP/ICMP → send it to detect() in detect\_intrusion.py.

This is the **real-time monitoring** part.

**Workflow of the Entire Project**

1. **Train model**  
   Run train\_model.py → produces a trained ML model (ids\_model\_simple.pkl).
2. **Start live IDS**  
   Run capture.py → starts sniffing packets.
   * Extract features → Encode → Predict → Detect → Log.
3. **When an attack is detected**  
   You’ll see messages like:

[2025-09-21 18:10:32] INTRUSION DETECTED! Features: [2, 1, 0, 300, 0]

and it gets logged in logs/intrusion\_logs.txt.

**Strengths**

* Simple & clear pipeline (data → train → detect).
* Uses **Random Forest** (good baseline model).
* Real-time sniffing with Scapy.
* Logs all intrusions for later analysis.