**Step-by-Step Guide for Implementing the IoT Network, IDS, and Attack Simulation Using Cooja**

**Step 1: Familiarize Yourself with Cooja**

Cooja is a simulation tool included in the Contiki OS. It supports various IoT devices and allows you to simulate IoT networks. Start by:

* Installing Contiki OS and Cooja on your system.
* Reviewing basic tutorials on setting up and running simulations in Cooja.

**Step 2: Design the IoT Network**

Use the GUI of Cooja to:

1. **Define Network Topology**:
   * Add nodes such as Tmote Sky or Z1 (as used in the paper).
   * Configure the network size (20, 30, 40 nodes, etc.).
   * Set distances between nodes (e.g., 20m, 30m, 40m).
   * Assign a border router for the connection to the IPv6 network.
2. **Configure Node Behavior**:
   * Install RPL (Routing Protocol for Low-Power and Lossy Networks) on the nodes.
   * Use Contiki's built-in RPL stack for communication.

**Step 3: Implement the IDS on Each Node**

The paper suggests an anomaly-based lightweight IDS implemented in a distributed manner:

1. **Code the IDS**:
   * Write a Contiki C program to monitor DIO and DIS messages.
   * Use thresholds based on normal behavior (e.g., average number of DIO messages + k×standard deviation).
2. **Deploy IDS**:
   * Compile the IDS code and upload it to each simulated node in Cooja.

**Step 4: Simulate Attacks**

The paper identifies two key attacks: Neighbor Attack and DIS Attack.

1. **Neighbor Attack**:
   * Modify the DIO behavior to simulate broadcasting false DIO messages.
2. **DIS Attack**:
   * Program nodes to flood the network with excessive DIS messages (either broadcast or unicast).

**Step 5: Test and Validate**

1. **Run Simulations**:
   * Observe node behavior under normal and attack scenarios.
   * Collect logs of message exchanges and node states.
2. **Analyze Results**:
   * Use True Positive Rate (TPR) and False Positive Rate (FPR) metrics to evaluate IDS performance, as done in the paper.

**Step 6: Document and Refine**

1. **Document Observations**:
   * Record detection accuracy, timing, and network overhead.
2. **Refine Thresholds**:
   * Adjust thresholds for anomaly detection to optimize performance.

**Tools and Resources**

* **Contiki OS and Cooja**: [Download and install from the official Contiki repository.](https://github.com/contiki-ng/contiki-ng)
* **Coding Environment**: Use an IDE like Eclipse with Contiki plugins or a text editor like VSCode.
* **Statistics and Logs**: Utilize tools like Wireshark (integrated with Cooja) to analyze network traffic.

**Suggested Next Steps**

* Implement specific algorithms from the paper, such as the threshold calculation formula for the IDS.
* Introduce variations in network size and attacker distribution (e.g., 1, 20%, or 30% of nodes as attackers) to test robustness.
* Validate against existing models like the one proposed by Lee et al. (compared in the paper).