Identifying DDoS Attacks in-Kernel via eBPF/XDP and Knowledge Distillation

Kezhuo Chen[1], Ding Yuan[2], Dehong Qiu*

School of Software Engineering, Huazhong University of Science and Technology, Wuhan 430074, China

(* Corresponding Author: Dehong Qiu, Email: qiudehong@hust.edu.cn)

Goals and challenges

The work has three core goals: First, resolve inefficiencies of user-space DDoS identification (e.g., heavy traffic load, delayed response) by building an in-kernel framework to process packets early in the Linux network stack. Second, use eBPF/XDP to enable high-speed packet interception/filtering in the NIC driver, ensuring low overhead and real-time detection. Third, design a lightweight yet accurate model via Knowledge Distillation (KD)—transferring knowledge from a high-performance MLP ("teacher") to a simple Decision Tree (DT, "student")—to fit kernel constraints while retaining MLP-level accuracy.

Key challenges include: (1) Adapting to strict kernel limits (limited memory, only simple instructions, no unbounded loops/decimals) that block complex models. (2) Balancing the DT's simplicity (for kernel deployment) and accuracy (matching MLP) when handling diverse DDoS types from CICIDS2017. (3) Optimizing kernel-compatible feature extraction: using recursive calculations for traffic features (e.g., Flow Duration stats) without storing excess packet data to avoid straining kernel resources.

Methods overview

1. In-kernel packet processing with eBPF/XDP

- Leverage XDP (eXpress Data Path) as a hook to attach eBPF programs to the network driver. This enables high-speed interception and filtering of incoming packets at the earliest stage of the Linux kernel network stack, reducing traffic load and response delays.

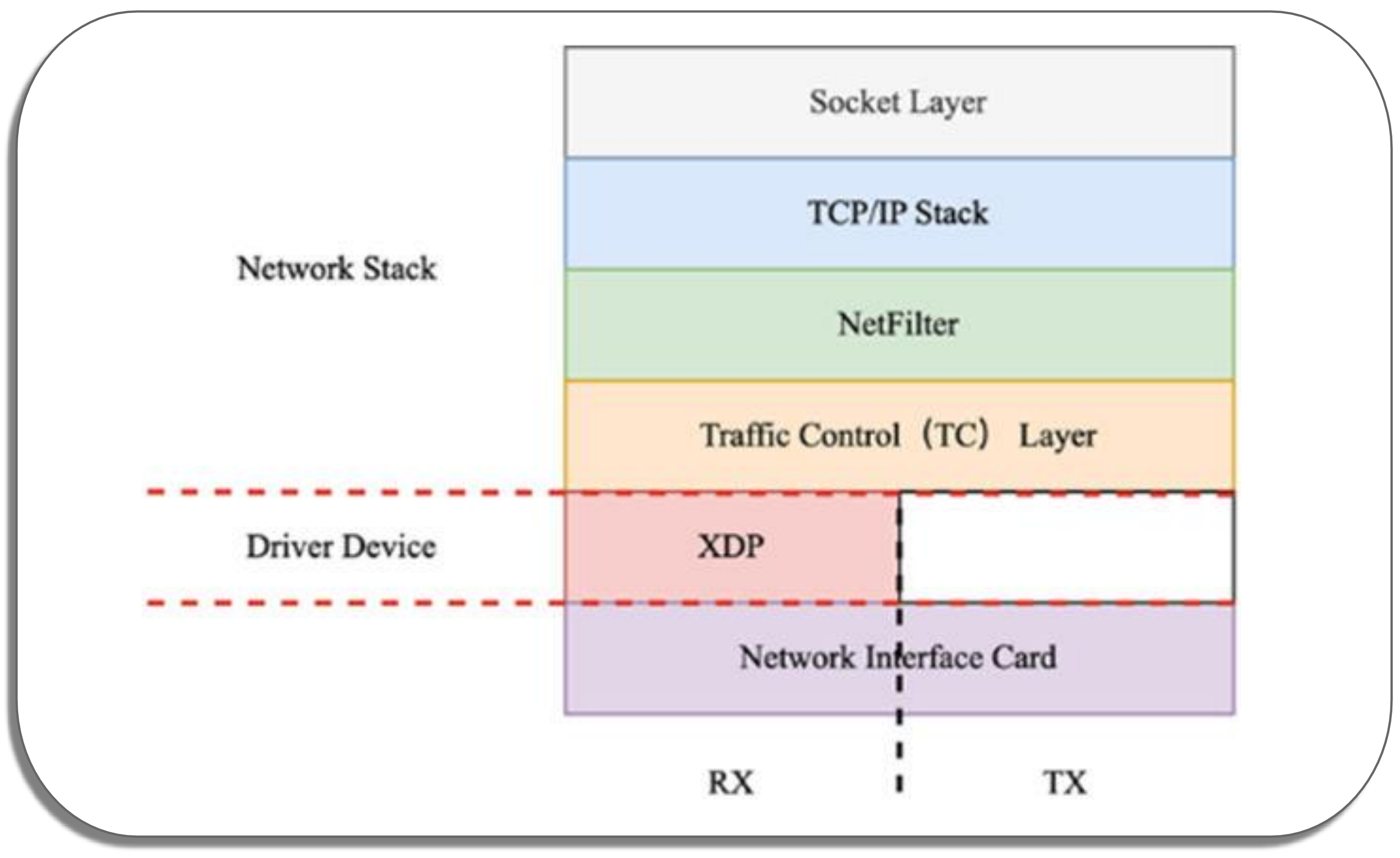


Fig. 1. Linux kernel network stack.

2. Feature extraction for traffic analysis

- Reassemble out-of-order arriving packets first, then compute 5 key traffic features (Flow Duration, Flow Rate, Packet Size, Flow Inter-Arrival Time, TCP flags). For the first 4 features, we calculate their Max, Min, Mean, and Variance via recursive formulas to ensure linear-time efficiency.

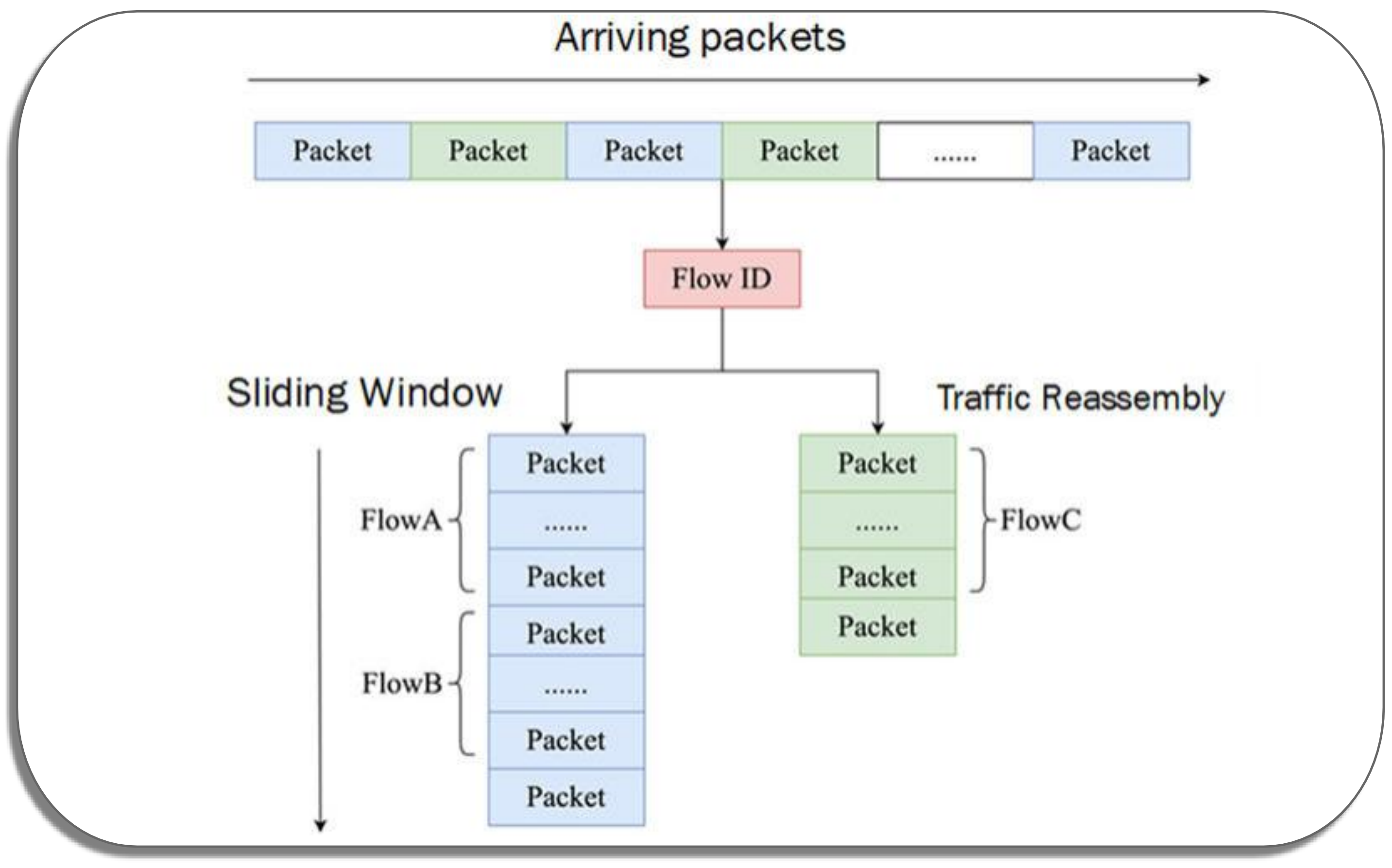


Fig. 2. Network traffic reassembly.

3. Knowledge Distillation (KD)-based model construction

- Train a complex MLP (teacher model) in user space using CICIDS2017 DDoS data to generate soft labels. Combine these with hard labels to form mixed labels, then train a lightweight CART (student Decision Tree, DT). Deploy the CART in the kernel via eBPF Maps (each node as a quintuple) to enable efficient DDoS identification under hardware constraints.

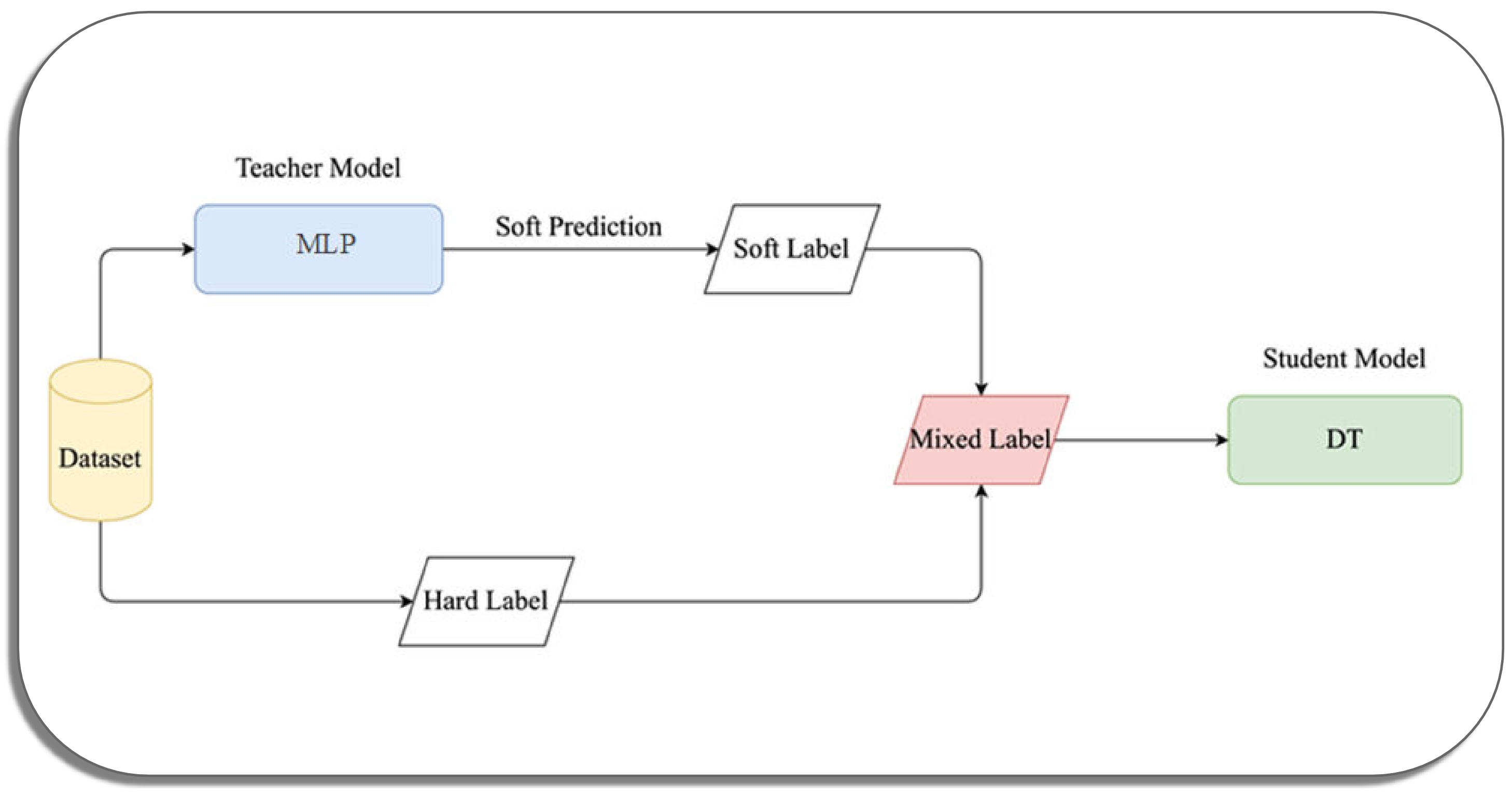
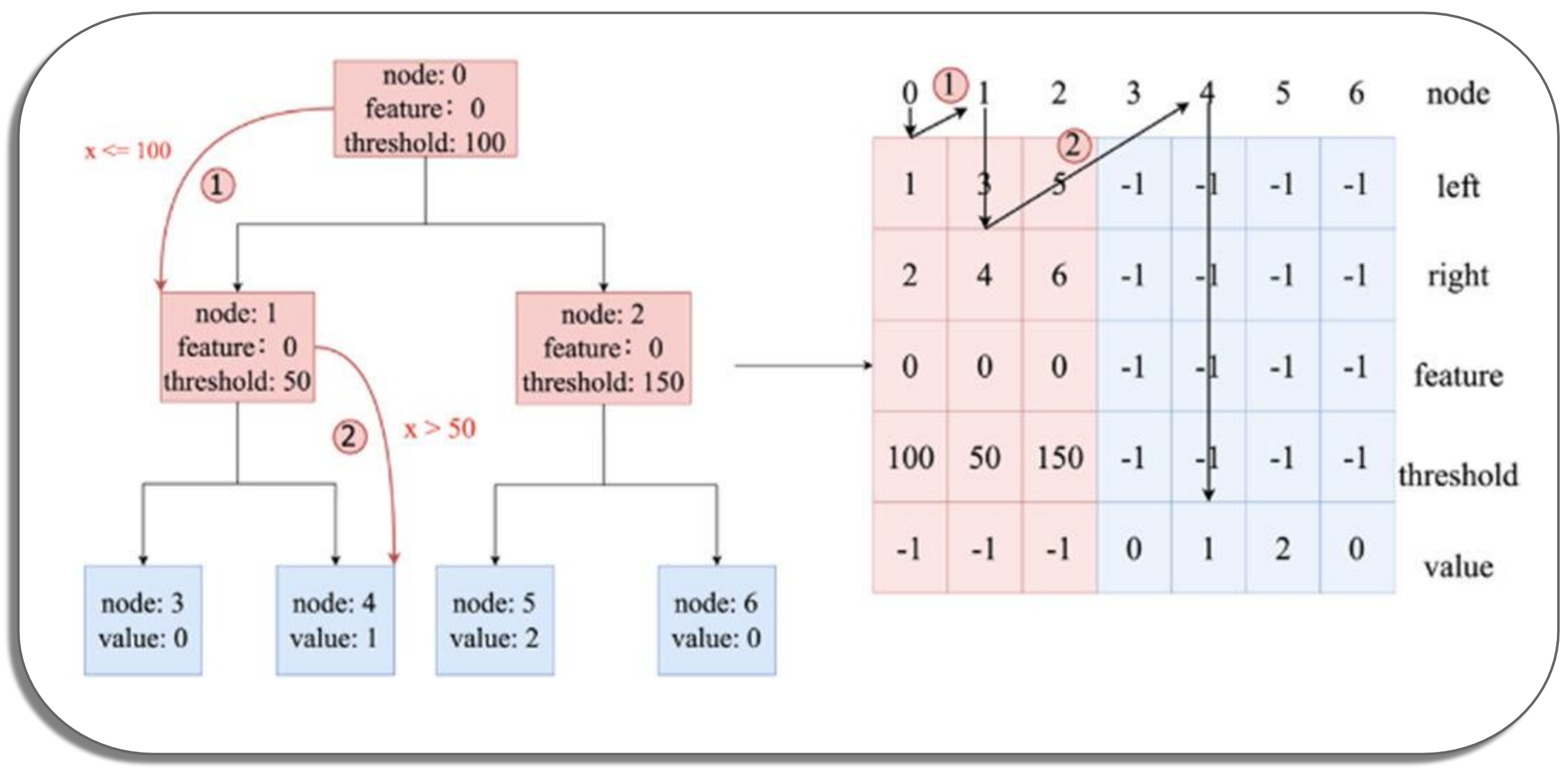


Fig. 3. The knowledge distillationfrom MLP to DT.



0	1	2	3	4	5	6	node
1	3	5	-1	-1	-1	-1	left
2	4	6	-1	-1	-1	-1	right
0	0	0	-1	-1	-1	-1	feature
100	50	150	-1	-1	-1	-1	threshold
-1	-1	-1	0	1	2	0	value

Fig. 4. The CART deployed in kernel.

Results

We evaluated our in-kernel DDoS identification model (DT-MLP) on the CICIDS2017 dataset (including benign traffic and 5 DDoS types: DoS Hulk, DDoS LOIT, DoS Slowhttptest, DoS Slowloris, DoS GoldenEye).

1.Identification Performance: DT-MLP achieved a Macro F1 Score of 97.6% — 1.1% higher than the baseline DT (96.5%) and nearly matching the teacher MLP (97.8%). It also maintained strong Macro Precision (0.966) and Macro Recall (0.987), proving knowledge from MLP was effectively distilled into DT.

2.Hyperparameter Impact: The optimal balance parameter (k=0.5) (for mixed labels) yielded the highest Macro F1 (97.6%); increasing k raised model complexity (node count rose from 296 at (k=0) to 1285 at (k=1)).

3.Model Depth Optimization: DT-MLP performed best with a maximum depth of 10–15—striking a balance between accuracy (stable Macro F1) and kernel memory efficiency (avoiding excessive complexity).

Future work

1.Deploy DT-MLP in real-world high-speed scenarios (e.g., data centers, cloud) to test its long-term stability and efficiency under actual traffic and complex attacks.

2.Enhance the kernel model via multi-source knowledge distillation (integrating CNN, GRU, etc., instead of a single MLP) to improve diverse DDoS identification.

3.Explore advanced compression (pruning, quantization) to reduce DT-MLP's kernel memory usage and instruction count for edge device deployment.

4.Extend the model to classify more DDoS subtypes in real time and integrate it with dynamic defense (e.g., adaptive filtering) for end-to-end in-kernel mitigation.