



第二届中国eBPF开发者大会

WWW.ebpftravel.com

基于eBPF的内核智能



中山大学 计算机学院 陈鹏飞 & 张钧宇

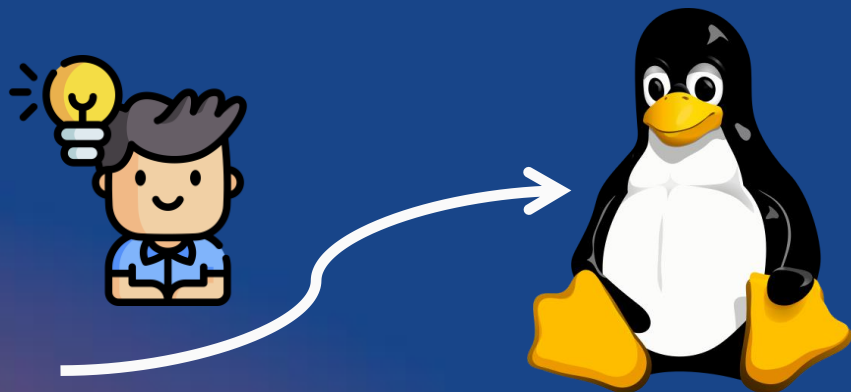
中国 · 西安

大纲

- 基于eBPF的内核智能
- 基于eBPF的实时神经网络的实现 (DSN24)
 - 动机：内核中实现神经网络的必要性和优势
 - 方案：低开销和高性能的神经网络的实现

内核智能

- 内核智能指的是在操作系统内核中加入智能化的程序，使其具备一定程度的自主学习、推理和决策能力，以增强系统在运行性能、安全防护和可观测性等方面的功能，同时不会破坏当前系统运行的稳定性和安全性



基于eBPF的内核智能

- eBPF 能够在 Linux 内核中完全动态和沙盒化地执行程序，无需对内核代码进行任何更改，为内核智能的实现提供了天然的生长土壤
- 目前已经有相关的探索方案：
 - Electrode[1]：利用eBPF实现分布式协议的消息广播等，提高协议的吞吐量
 - DINT[2]：利用eBPF将频繁事务操作卸载到内核，降低网络栈带来的通信开销

[1] Zhou, Y., Wang, Z., Dharanipragada, S. and Yu, M., 2023. Electrode: Accelerating Distributed Protocols with eBPF. In 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23) .

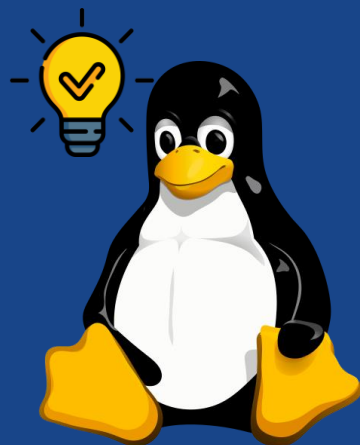
[2] Zhou, Y., Xiang, X., Dharanipragada, M.K.S. and Yu, M., 2023. DINT: Fast In-Kernel Distributed Transactions with eBPF. Yang Zhou, p.1.



第二届中国eBPF开发者大会

WWW.ebpftravel.com

进一步的智能：让内核拥有自我思考的能力？



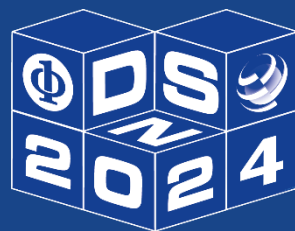
中国 · 西安



第二届中国eBPF开发者大会

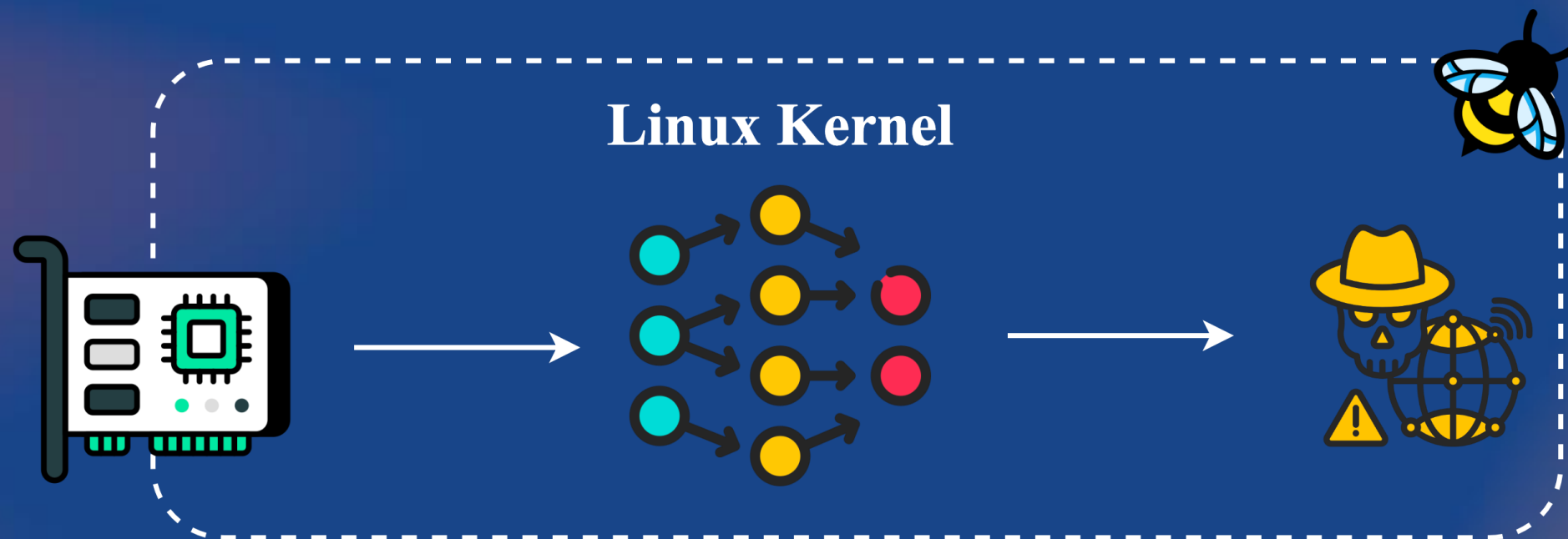
WWW.ebpftravel.com

基于eBPF的实时 神经网络的实现



中国 · 西安

eBPF + Neural Network in Kernel = ?



Why ?

实时网络检测和防御

- 公有云服务的剧增强调了实时网络检测和防御的重要性
- 运行在用户态的模型忽略了实时网络测量的开销
- 基于规则的主动防御工具需要在查询开销和防御等级之间取舍

Maintain a secure environment with security tools and cloud security software in AWS Marketplace


Whether you are securing endpoints, identifying vulnerabilities, or safeguarding sensitive data, you can find the security software and security tools you need on AWS Marketplace to enhance protection for your entire Amazon Web Services (AWS) environment with compatible security solutions.

Benefits

- Deploy a comprehensive security architecture**
From initial migration through ongoing day-to-day security platform management, leverage independent software vendors (ISVs) with proven success securing cloud adoption.
- Reduce risk without losing speed**
Minimize business disruptions by quickly procuring and deploying enterprise security software solutions that can manage identity and access, detect intrusions, and enable faster response times.
- Integrate security tools**
Leverage security tools designed for AWS to integrate with your existing security best practices.

on-demand webinar

Learn how SOAR helps you streamline security while improving your defenses against cyber attacks



云堡垒机 CBH

云堡垒机 (Cloud Bastion Host) 开箱即用，包含主机管理、权限控制、运维审计、安全合规等功能。支持Chrome等主流浏览器随时随地远程运维，开启高效运维新时代

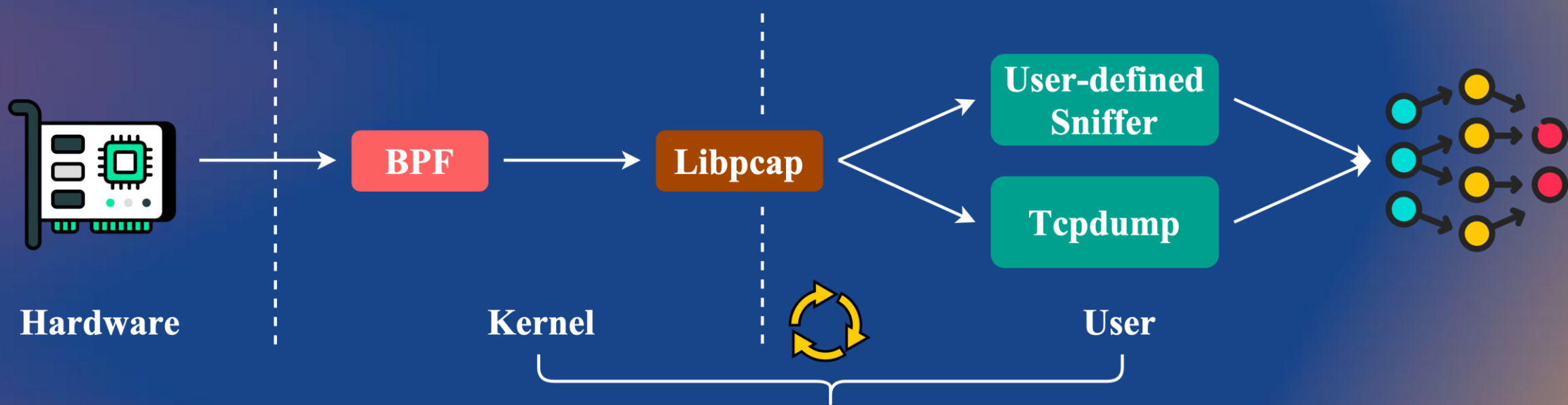
[立即购买](#) [管理控制台](#) [帮助文档](#)

了解云堡垒机 CBH

- 操作审计**
全程记录用户在系统的操作行为，审计用户对目标资源的操作，识别运维风险
- 权限管控**
集中管控用户访问系统和资源的权限，对系统和资源的访问权限进行细粒度设置
- 高效运维**
支持Web浏览器运维、自动化运维，第三方客户端运维，满足等保合规要求，提供身份认证、接入多种运维工具



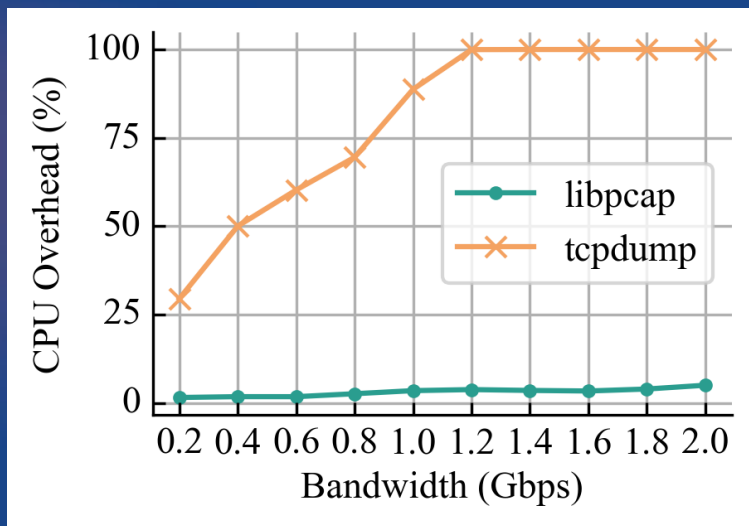
基于用户态的实时网络检测



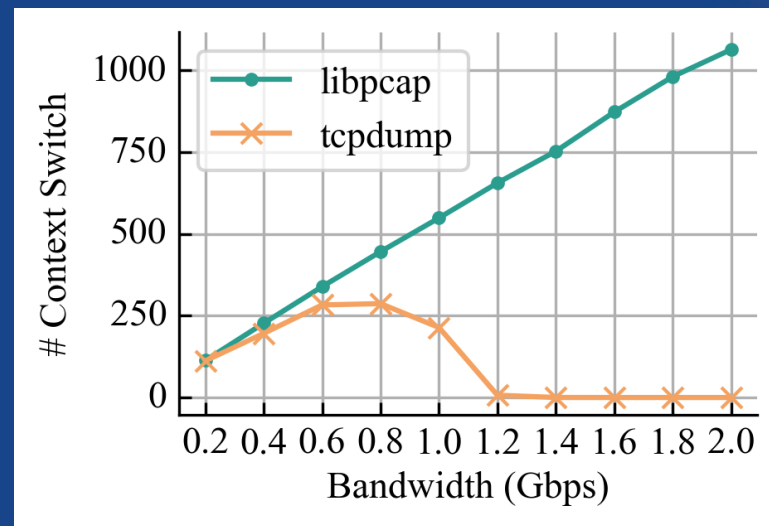
问题：大量的上下文切换开销



问题：用户态实时抓包工具的开销



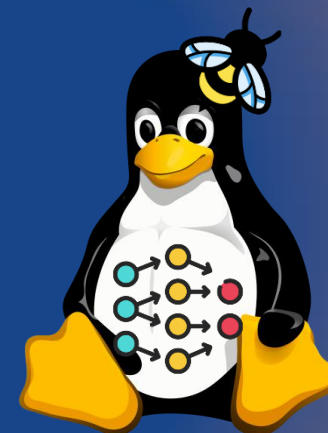
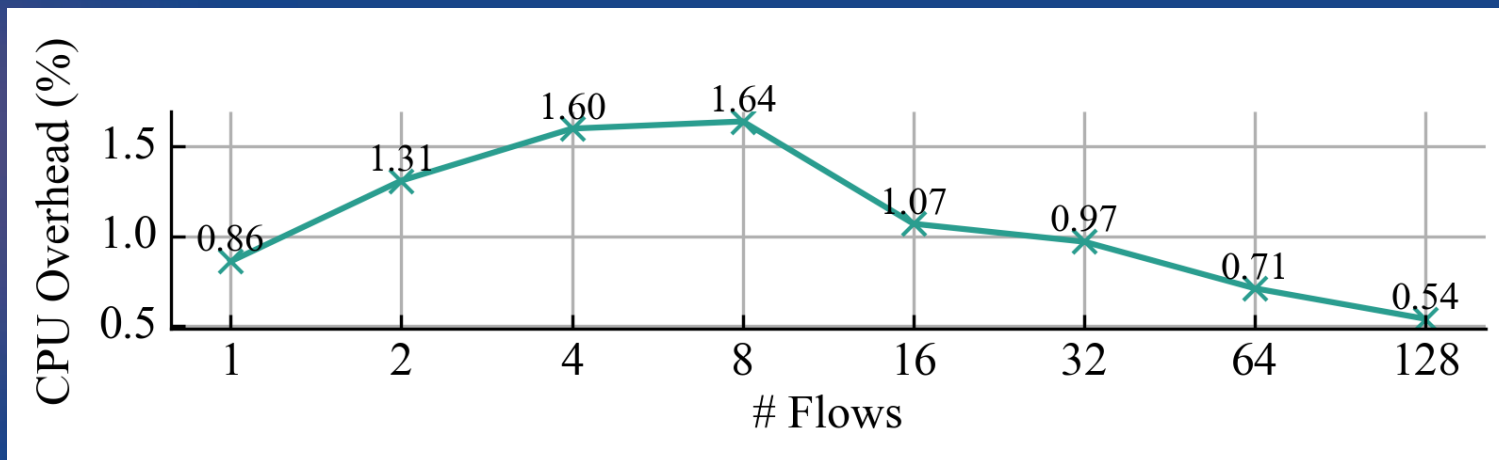
CPU开销



上下文切换开销

- CPU: tcpdump (29-100%), libpcap (2-5%)
- 上下文切换：可达每秒1k次，和带宽成线性关系，最终导致丢包

eBPF + Neural Network in Kernel = 低开销



- 测量条件：1-128个并发流，均以最大速率传输
- CPU开销：0.54-1.64%，包含抓包、特征提取和神经网络推理
- 内存开销：5KB，包括神经网络参数等

基于eBPF的实时网络测量

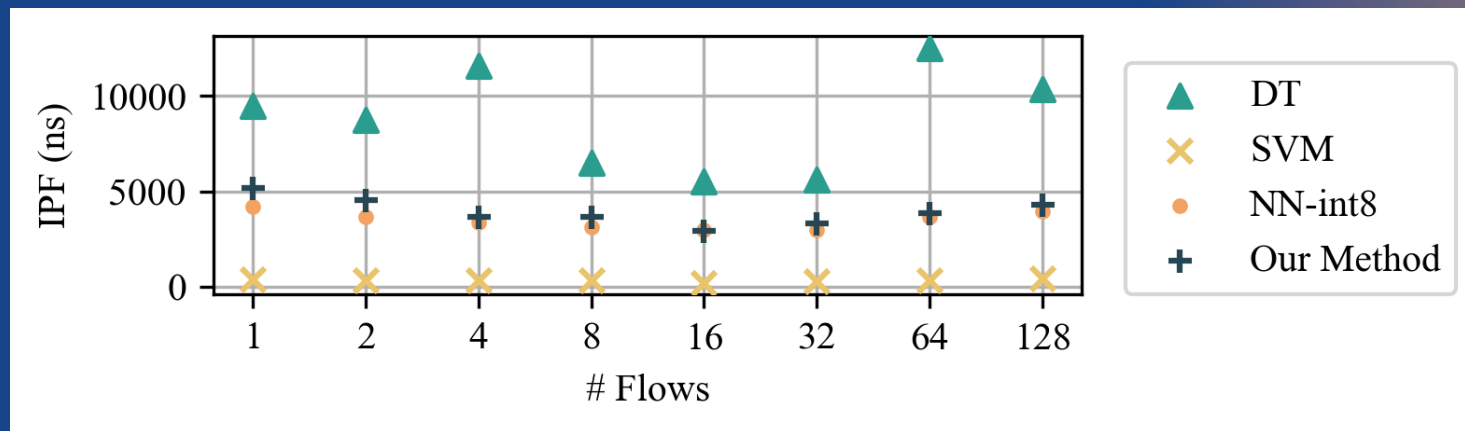
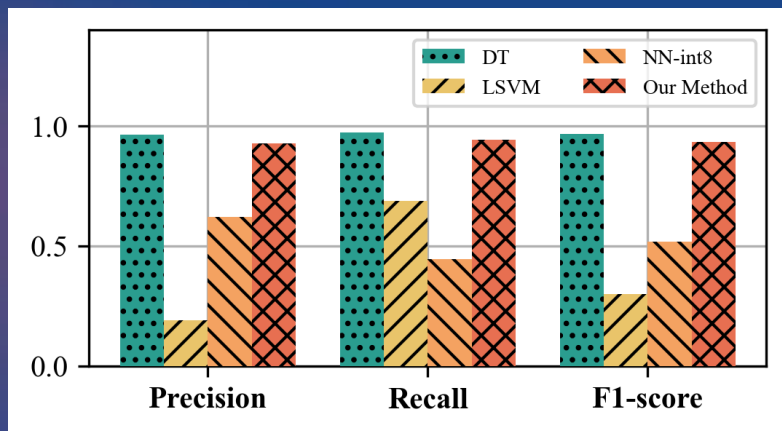
- eBPF 能够在 Linux 内核中动态和沙盒化地执行用户定义程序
- 类似的方案 (eBPF + ? in kernel) :
 - 决策树 [1] (Decision Tree, DT)
 - 线性支持向量机 [2] (Linear Support Vector Machine, LSVM)
 - 基于int8量化的神经网络 [3] (Neural Network with int8 Quantization, NN-int8)

[1] Maximilian Bachl, Joachim Fabini, and Tanja Zseby. A flow-based ids using machine learning in ebpf. arXiv preprint arXiv:2102.09980, 2021.

[2] NEMALIKANTI ANAND, MA SAIFULLA, and Pavan Kumar Aakula. High-performance intrusion detection system using ebpf with machine learning algorithms. 2023.

[3] Takanori Hara and Masahiro Sasabe. On practicality of kernel packet processing empowered by lightweight neural network and decision tree. In 2023 14th International Conference on Network of the Future (NoF), pages 89–97. IEEE, 2023.

eBPF + NN in Kernel = 低开销 + 高性能



- 案例：入侵检测
- DT：指数级的内存开销，实时检测时间（IPF）不确定
- SVM：拟合能力不足
- NN-int8: int8量化显著降低模型精度

被当前工作忽略的挑战

➤ 概念漂移 (Concept Drift)

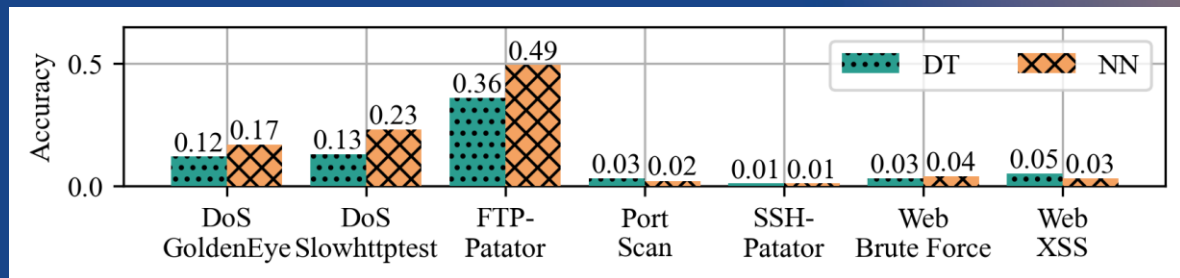
- 在线模型产生关键性错误
- 更新时保证安全 (竞争条件+热更新)

➤ eBPF的编程限制

- 1M 指令数量
- 512B 栈空间: 选择最重要的特征

➤ 竞争条件

- 多核执行eBPF程序
- eBPF Spin Lock的限制和开销



概念漂移

- When the lock is taken, calls (either BPF to BPF or helpers) are **not allowed**.
- The `BPF_LD_ABS` and `BPF_LD_IND` instructions are **not allowed** inside a spinlock-ed region.
- The BPF program MUST call `bpf_spin_unlock()` to release the lock, on all execution paths, before it returns.
- The BPF program can access `struct bpf_spin_lock` only via the `bpf_spin_lock()` and `bpf_spin_unlock()` helpers. Loading or storing data into the `struct bpf_spin_lock lock`; field of a map is **not allowed**.
- To use the `bpf_spin_lock()` helper, the BTF description of the map value must be a struct and have `struct bpf_spin_lock anyname`; field at the top level. Nested lock inside another struct is **not allowed**.

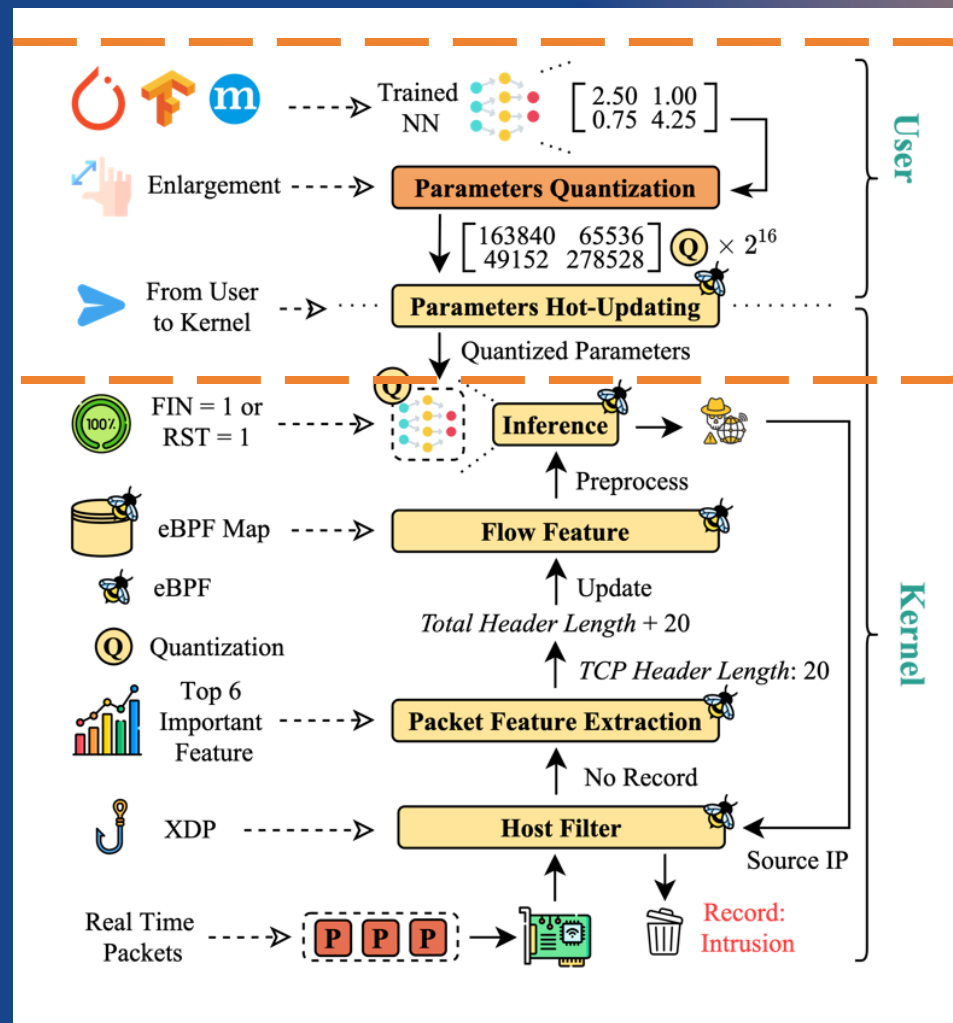
我们的方案

➤ 用户态训练和量化 (NN Training & Parameters Quantization) :

- 训练: 利用Pytorch等框架训练NN
- 量化: 将模型参数从浮点数转换成整数

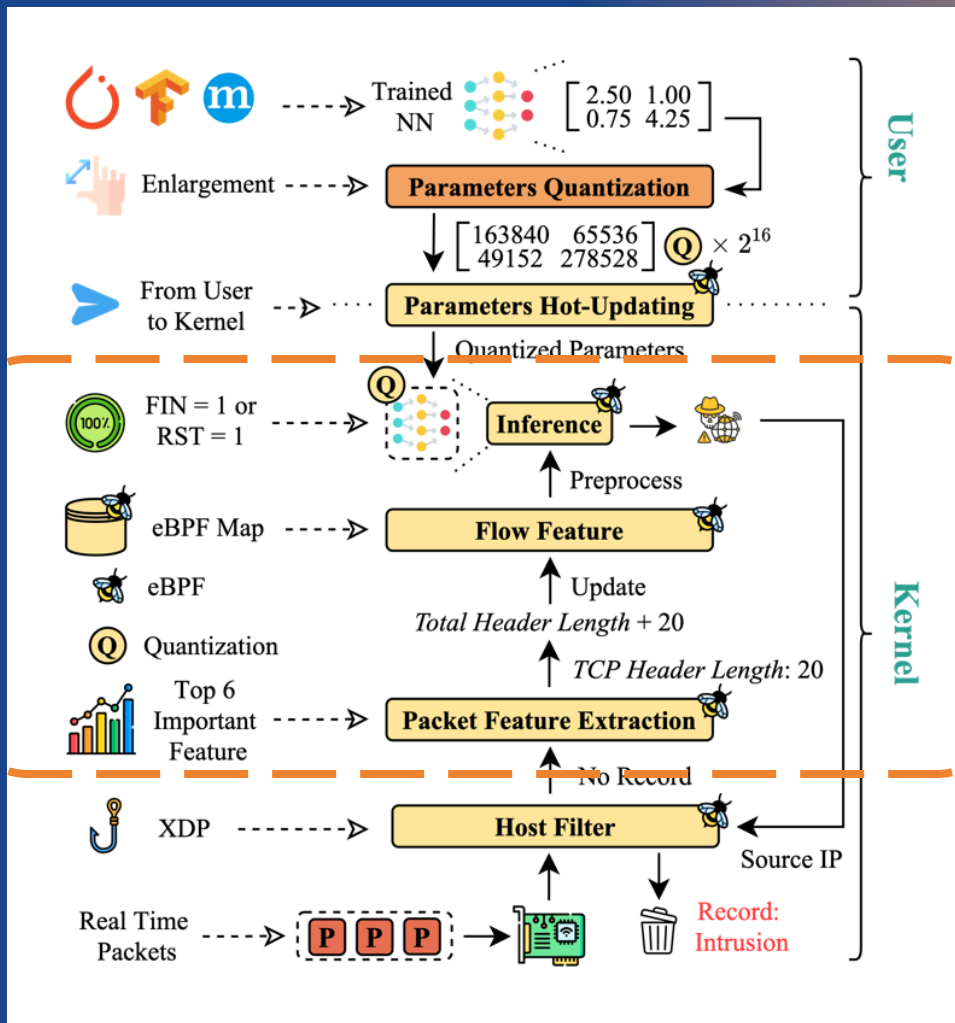
➤ 参数热更新 (Parameters Hot Updating) :

- 解决了内核态-用户态热更新参数的读写竞争条件问题



我们的方案

- 特征提取和流特征更新 (Packet Feature Extraction & Flow Feature) :
 - 实时提取packet特征并更新flow的特征
- NN推理 (Inference) :
 - Flow结束后根据其特征进行推理
 - 仅使用整数实现



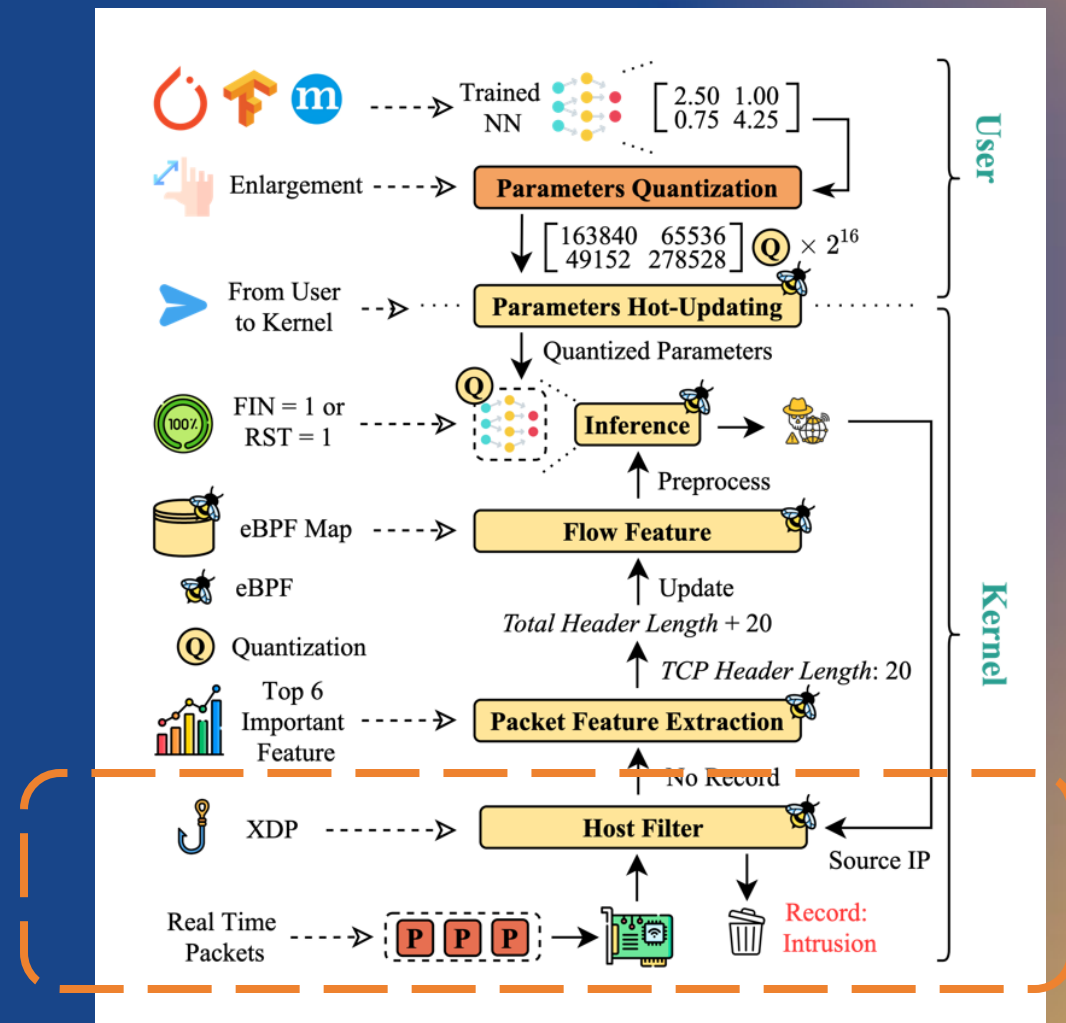
我们的方案

➤ 过滤器(Host Filter):

- 丢弃来自之前识别为进行异常的主机的数据包

➤ 异常主机记录与纠正:

- NN检测结果被记录到Host Filter中
- 人工干预修正



➤ 目前实现的是Multilayer Perceptron (Linear + ReLU)

➤ eBPF不支持浮点数运算，需要对推理过程进行量化：

- 基本思想：使用定点数的方式存储浮点数
- 模型参数的离线量化： $W_E^{(k)} \triangleq \text{round}(s \cdot W^{(k)}) = \left[\text{round}(s \cdot w_{ij}^{(k)}) \right]$
- 预处理的实时量化：eBPF不支持有符号数除法

神经网络推理
实现方案

$$x_j^{(1)} = \begin{cases} 0, & \sigma_j = 0 \\ -\text{round}\left(\frac{s \cdot (\mu_j - x_j)}{\sigma_j}\right), & x_j < \mu_j, \sigma_j \neq 0 \\ \text{round}\left(\frac{s \cdot (x_j - \mu_j)}{\sigma_j}\right), & x_j > \mu_j, \sigma_j \neq 0 \end{cases}$$

➤ 推理过程的实时量化:

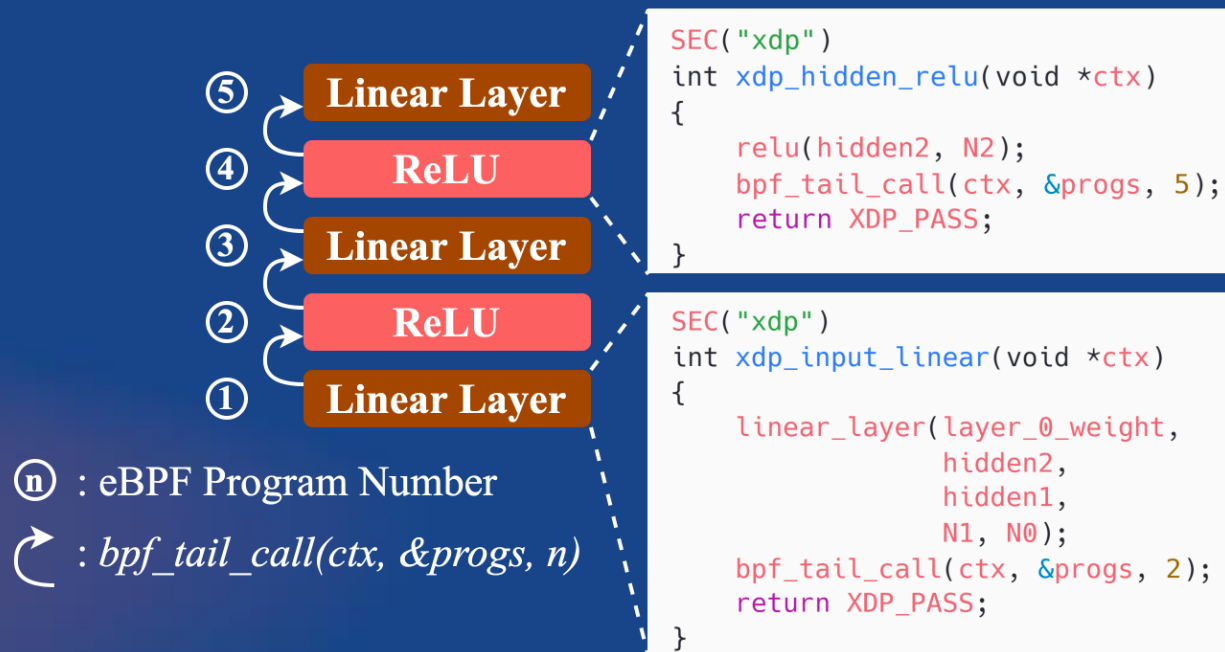
- Linear Layer: 当 $s = 2^b$ 时, 除法可以通过逻辑右移 b 位来实现

$$\mathbf{y}^{(k)} = \frac{1}{s} \cdot \mathbf{W}_E^{(k)} \cdot \mathbf{x}^{(k)} = \frac{1}{s} \cdot \left[\sum_{j=1}^{n^{(k-1)}} w_{E,ij}^{(k)} \cdot x_j^{(k)} \right]$$

- ReLU:

$$\mathbf{x}^{(k)} = \text{relu}(\mathbf{y}^{(k-1)})$$

神经网络推理
实施方案

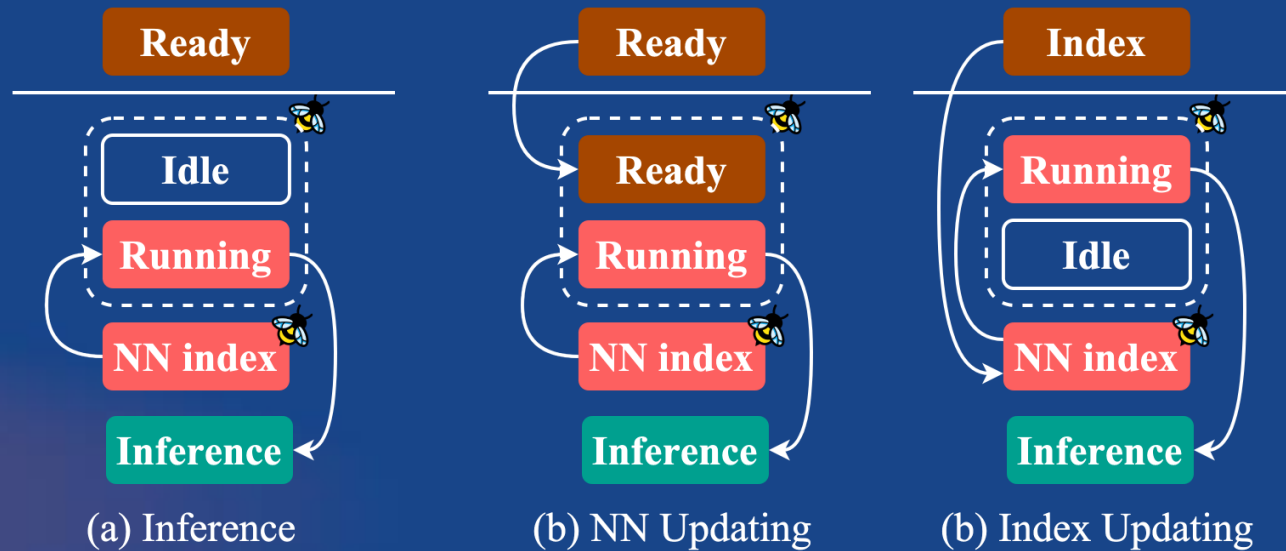


$$x^{(k)} = \text{relu}(y^{(k-1)})$$

$$y^{(k)} = \frac{1}{s} \cdot W_E^{(k)} \cdot x^{(k)}$$

实现方案
突破指令数的限制

- [6,32,32,2]的三层MLP被eBPF verifier拒绝加载
- 基本思想:
 - MLP = Linear Layer + ReLU的链式组合
 - 每一层使用一个eBPF program表示, 使用Tail Call调用下一层
- NN的深度受到最大Tail Call层数 (33) 的限制



➤ 无显式eBPF Spin Lock解决热更新时的读写竞争条件

➤ 基本思想：两阶段加载

- NN Updating: 加载新的神经网络参数 (Ready) 到内核的闲置存储空间 (Idle)
- Index Updating: 更新正在使用的参数 (Running) 的指针 (NN Index), 使其指向Idle

实现方案
神经网络参数的热更新

- 两个常用的Hook点：XDP和TC
- eXpress Data Path (XDP)：只能过滤RX数据包
- Traffic Control (TC)：
 - 在XDP之后触发，可以处理RX和TX数据包
 - 额外的内存分配和socket队列等待的开销，性能上不如XDP
- 选用XDP作为eBPF挂载点
 - 利用DT给出特征重要性排序
 - 选取Top 6个特征的提取算法在XDP中实现 (Destination Port, Total Forward Packets, Total Length of Forward Packets, Forward Packets Length {Min, Max, Std})

实现方案
eBPF挂载点的选取

结论

- 内核智能的实现有效增强了当前内核的能力
- eBPF技术给内核智能的实现注入了强大活力
- 目前关于基于eBPF的内核智能的探索方兴未艾



第二届中国eBPF开发者大会

WWW.ebpftravel.com

谢谢大家!



中国·西安