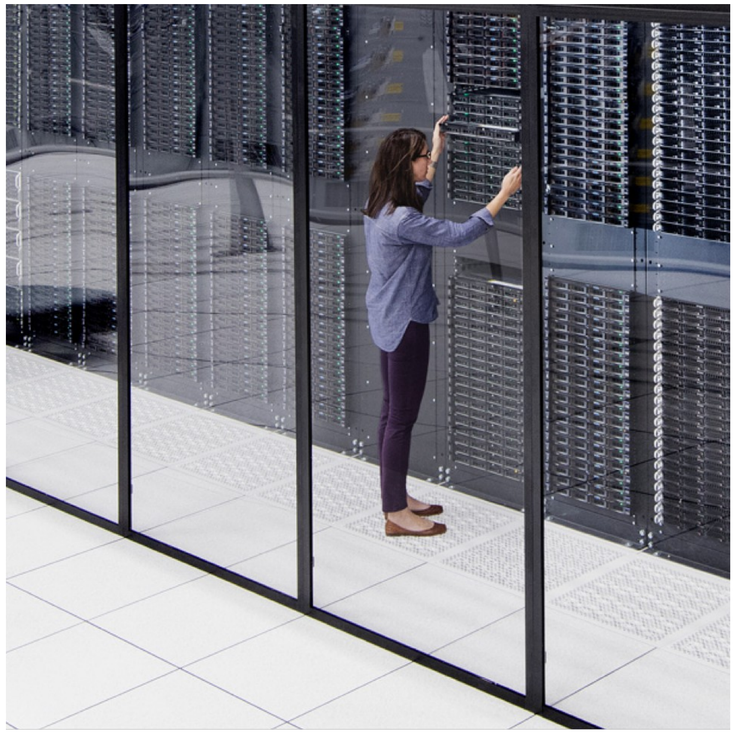
ML applications in Linux kernel BoF

Viacheslav Dubeyko
PhD, Linux kernel developer
IBM



ML in Linux kernel

- The number of areas of **Machine Learning (ML)** approaches application is **growing** with every day.
- There are already research works and industry efforts to employ **ML approaches in Linux kernel**.
- The using of ML approaches in Linux kernel is **inevitable step**.
- ML approaches can help to make Linux kernel development more productive and reliable.
- ML approaches can help to execute more exhaustive code testing and review.
- ML approaches can help to optimize Linux kernel performance and efficiency.

However, **running ML model(s)** in Linux kernel is **not straightforward and easy** direction:

- Mostly, there is no direct use of floating-point operations (FPU) in Linux kernel code.
- Training phase cannot be executed on kernel side because of performance degradation penalties.

Key questions

1) How can we use ML approaches in Linux kernel?
2) Which infrastructure do we need to employ in Linux kernel for adoption of ML methods?
3) How can we use ML to automate testing, bugs detection, and bugs isolation in Linux kernel?
4) Can we use ML for automated refactoring and bug fix in Linux kernel code?
5) Can we use the whole fleet of Linux kernel deployments for massive and distributed testing of Linux kernel functionality?
6) How can we optimize the Linux kernel by using ML techniques?
7) How to make ML techniques working in Linux kernel without affecting performance and efficiency the Linux kernel operations?

Directions

- Using AI/ML for Linux kernel code management
- Using ML for kernel testing and bug fix
- Using ML techniques for efficient task scheduling policies elaboration
- ML infrastructure in Linux kernel
- Using ML Techniques + eBPF for achieving efficient kernel configuration
- Using ML methods for enhancing file systems reliability
- Using ML methods for prediction storage drives failures
- Using ML methods for memory access patterns analysis
- Using ML methods for delta-encoding and decreasing WAF in file systems
- Using ML methods for FSCK and file systems synthesis
- Using ML methods for efficient data placement policy
- Using ML methods for CXL benchmarking
- Using ML for re-writing the whole Linux kernel in Rust
- Cognitive file systems: using ML for data classification and data organization
- Using hardware acceleration for employing ML methods on kernel side

ML infrastructure in Linux kernel

Two important trends:

1) moving kernel-space functionality into user-space drivers (for example, SPDK, DPDK, ublk).

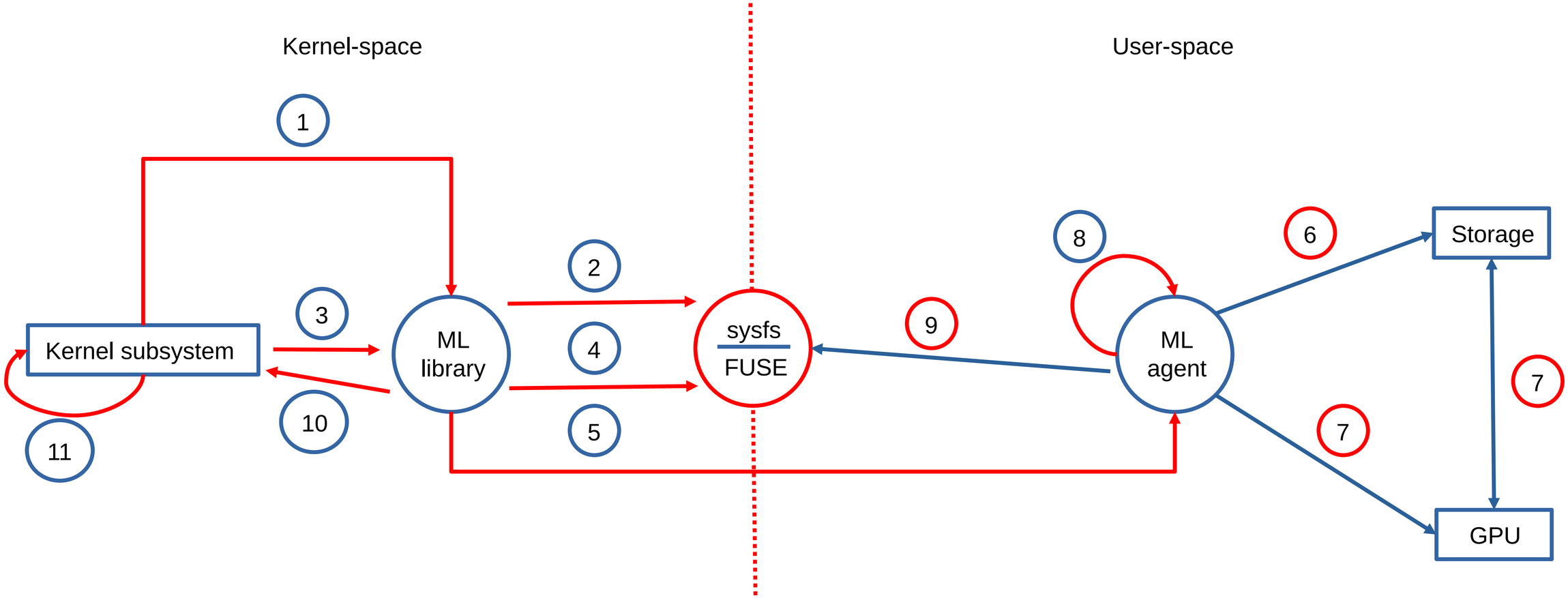
2) significant number of efforts of using ML models for various real life applications (for example, tuning kernel parameters, storage device failure prediction, fail slow drive detection, and so on).

Problem: direct implementation of ML approaches in Linux kernel-space is very hard, inefficient, and problematic because of practical unavailability of floating point operations in the Linux kernel, and the computational power hungry nature of ML algorithms (especially, during training phase).

Solution:

- Both cases (user-space driver and ML subsystem) require a user-space functionality that can be considered as user-space extension of Linux kernel functionality.
- The key responsibility of kernel-side agent (or subsystem) is the accounting of user-space extensions, synchronization of their access to shared resources or metadata on kernel side, statistics gathering and sharing it through the sysfs or specialized log file (likewise to syslog).
- Such specialized log file(s) can be used by ML user-space extensions for executing the ML algorithms with the goal of analyzing data and available statistics.
- This ML logic can elaborate some “recommendations”, for example, that can be shared with an ML agent on the kernel side.
- The kernel-space ML agent can check the shared “recommendations” and to apply the valid “recommendations” by means of Linux kernel tuning, recompilation, “hot” restart and so on.

ML infrastructure in Linux kernel



1. Declare kernel subsystem parameters
2. Create sysfs entries
3. Periodically update values of the parameters
4. Publish values in sysfs
5. Notify user-space ML agent

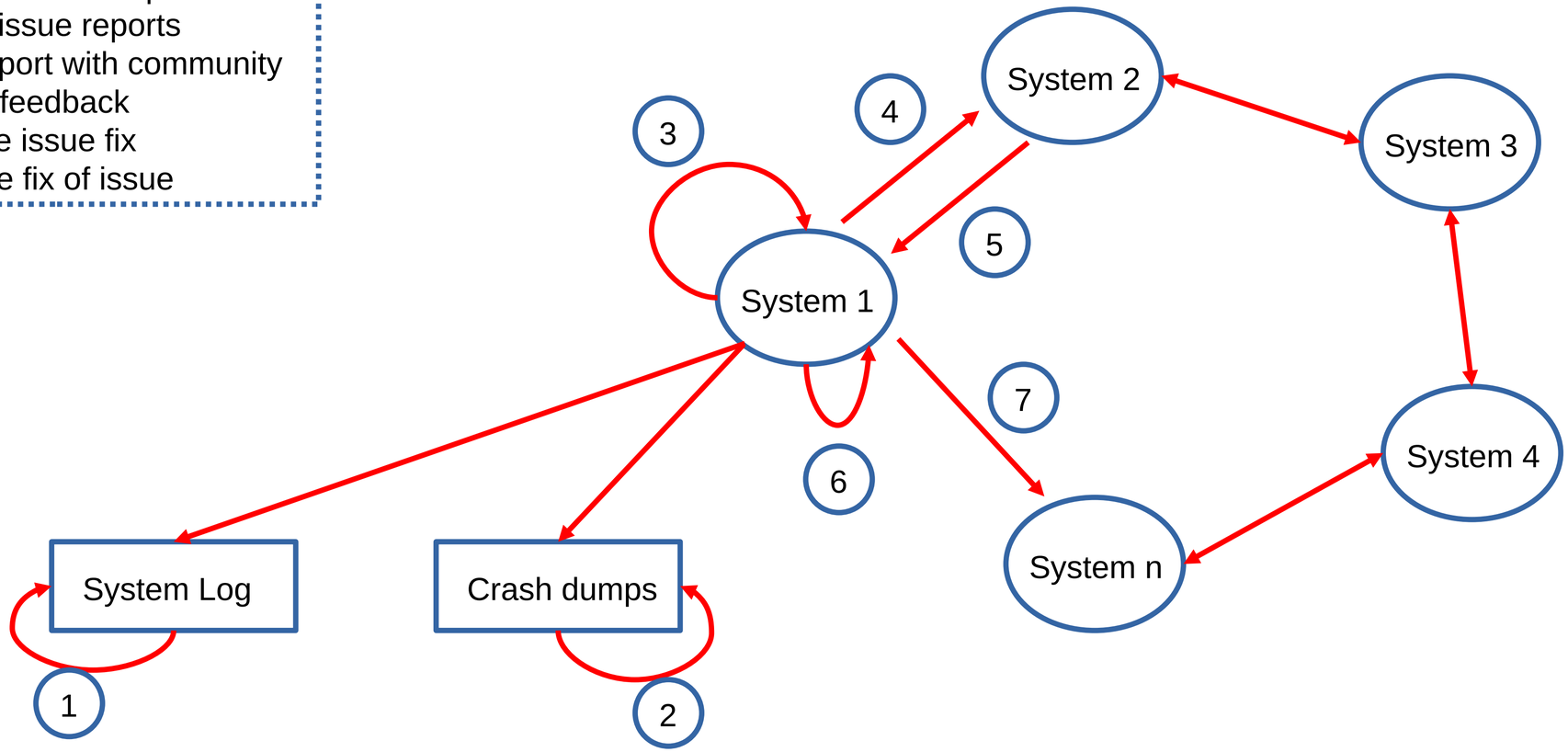
6. Save knowledge into storage
7. Train ML model for available data
8. Elaborate ML model
9. Share “recommendations” with kernel-space
10. Notify kernel subsystem
11. Apply “recommendations” on kernel side

ML for kernel testing and bug fix

- Issue reproduction script generation
- Issue reproduction + fix
- Code review + refactoring
- Unit-tests generation
- Automated research and statistics gathering on the whole fleet of running Linux kernels
- Automated reporting about bugs and crashes

ML for kernel testing and bug fix

1. Analyze system log
2. Analyze crash dumps
3. Prepare issue reports
4. Share report with community
5. Receive feedback
6. Elaborate issue fix
7. Share the fix of issue



ML methods for memory access patterns analysis

Architecture:

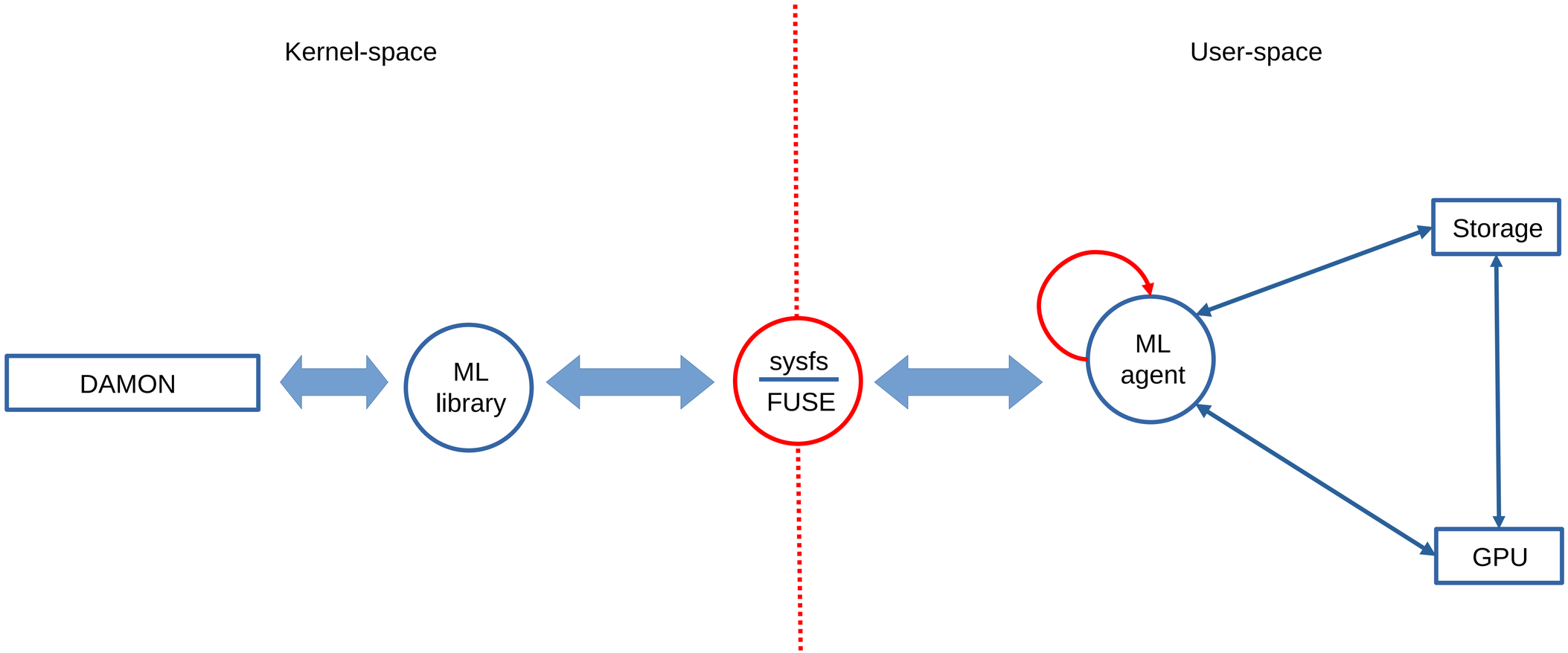
- DAMON already has sysfs interface with user-space
- User-space thread(s) can run ML model(s) and to interact with kernel space
- User-space threads can periodically receive statistics, interact with kernel space, and execute the neural network training or/and inference phase

Which ML type(s) can be used?

- The **supervised ML** model means that somebody (or something) can prepare memory access patterns and to assign some classification labels to them
- Neural network can be trained on this memory access patterns dataset and it can try to find these memory access patterns (bad or good) during inference phase

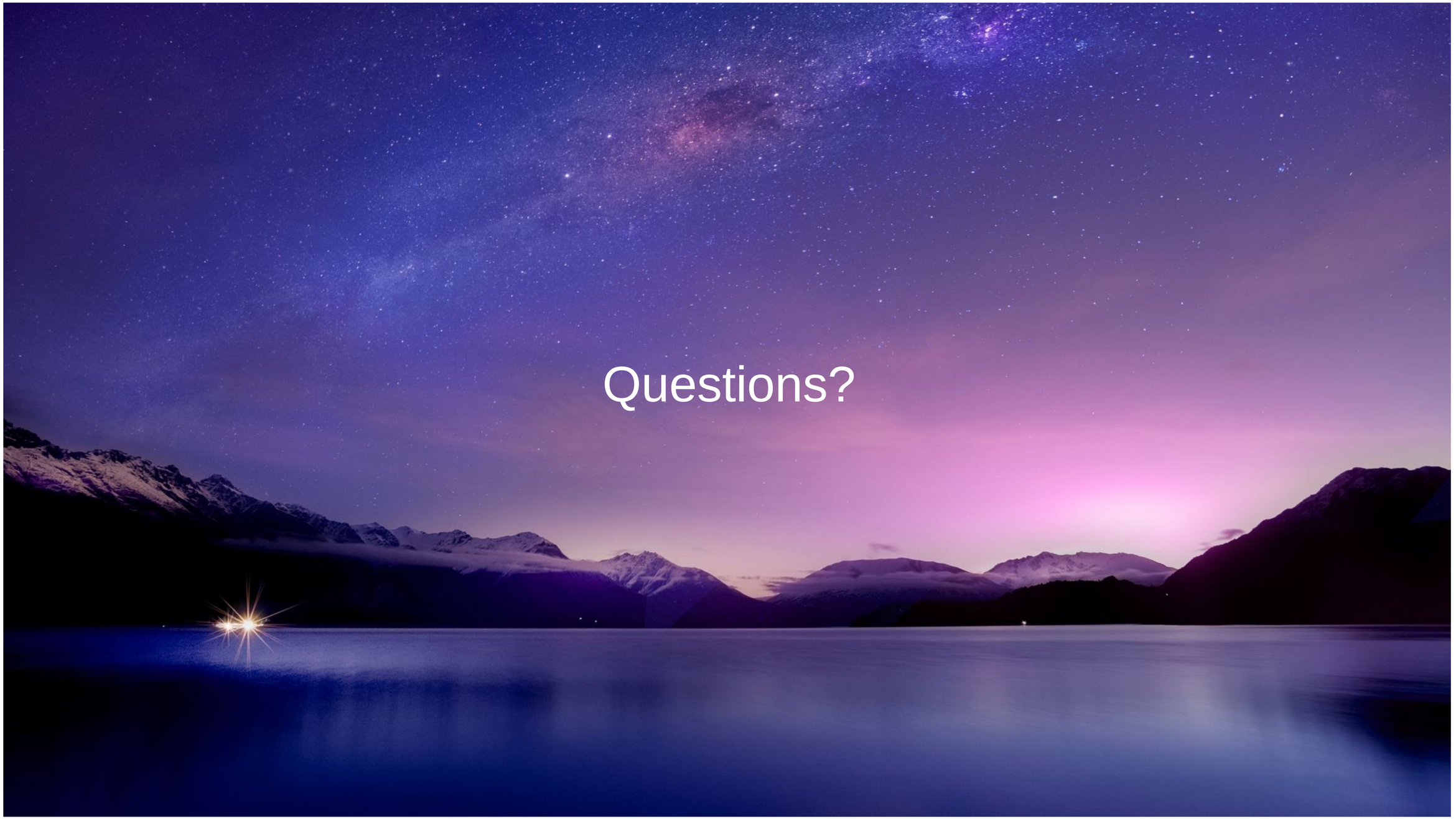
- The **unsupervised ML** model implies that we haven't such memory access patterns dataset
- ML model can try to find the repeatable patterns in statistics and, then, it could try to identify usual reproducible patterns and to detect outliers.

DAMON + ML infrastructure in Linux kernel



Discussion

- ML infrastructure in Linux kernel
- ML for kernel testing and bug fix
- ML methods for memory access patterns analysis
- Any other topic????


Questions?