

Progress Report

Objective 1-1: Understand the eBPF runtime mechanism for both ARM and x86 platforms.

Objective 1-2: Integrate eBPF hooks with existing HPDF data streaming applications.

Objective 2-1: Encapsulate existing HPDF-H containerized applications into Kubernetes services.

Objective 2-2: Deploy the Kubernetes control plane on "nvidarm" and offload its network functions.

Objectives		Telemetry					Kubernetes			
		Obj 1-1	Obj 1-2	Obj 1-3	Obj 1-4	Obj 1-5	Obj 2-1	Obj 2-2	Obj 2-3	Obj 2-4
FY25	Q1 (Oct-Dec 2024)									
	Q2 (Jan-Mar 2025)									
	Q3 (April-June 2025)									
	Q4 (July-Sep 2025)									



Let's check if the service was created.

```
try:
    stdout, stderr = node1.execute("kubectl get service kubernetes-bootcamp")
    print(f"stdout: {stdout}")
    print(f"stderr: {stderr}")
except Exception as e:
    print(f"Exception: {e}")
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes-bootcamp	ClusterIP	10.111.180.18	<none>	8080/TCP	11s

```
$ sudo cat /sys/kernel/debug/tracing/trace_pipe
```



```
node-1254811 [007] ..s1 8737831.671074: 0: Got IP packet: tot_len: 79,
sshd-1254728 [006] ..s1 8737831.674334: 0: Got IP packet: tot_len: 79,
sshd-1254728 [006] ..s1 8737831.674349: 0: Got IP packet: tot_len: 72,
node-1254811 [007] ..s1 8737831.674550: 0: Got IP packet: tot_len: 71,
```

TELEMETRY

- Reproduce the below tutorial on a x86 RHEL Linux Kernel 4.8 machine (my labtop): <https://eunomia.dev/en/tutorials/20-tc/#compilation-and-execution>;
- Most recent SW on "nvidarm".

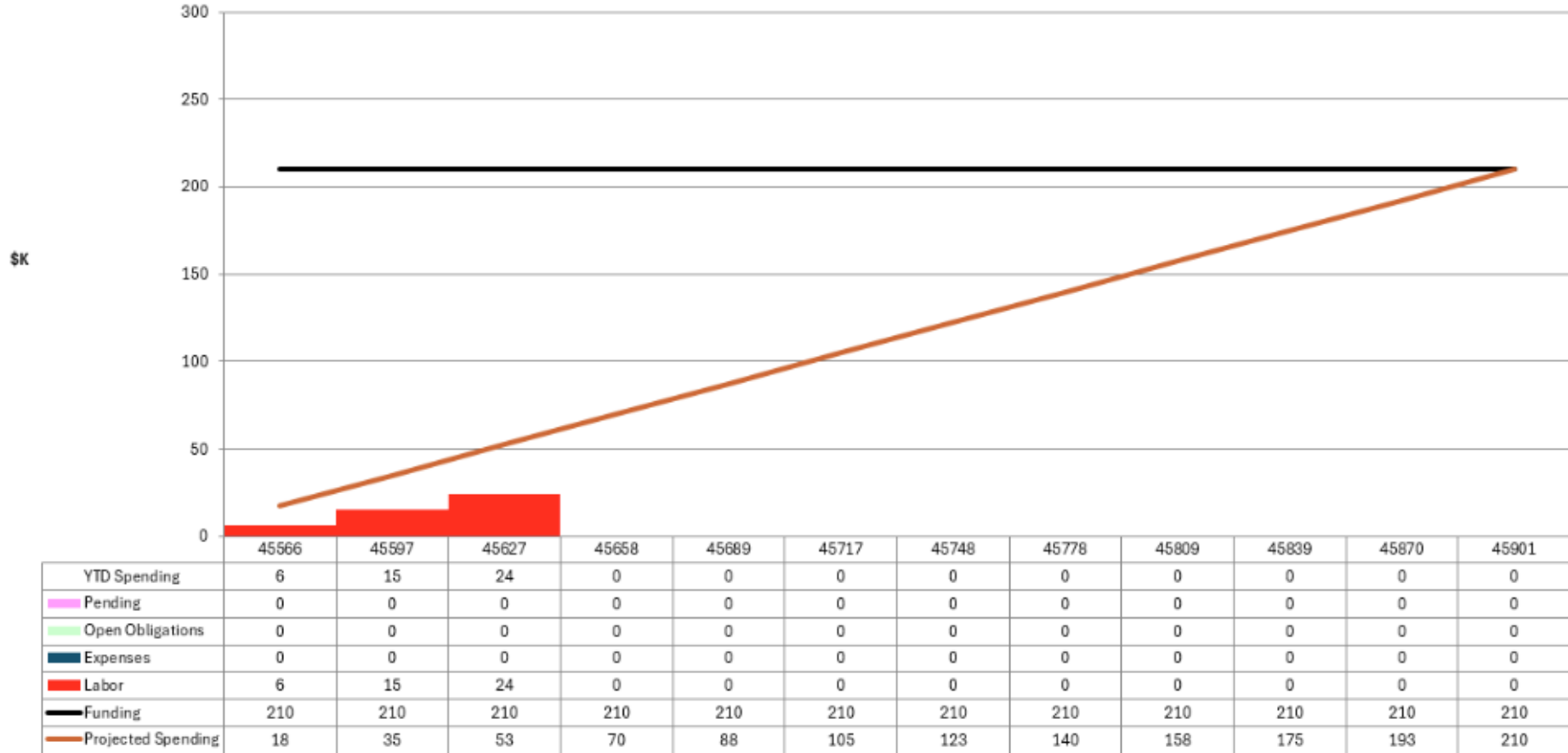
Fabric Testbed Notebooks:

- https://github.com/cissieAB/DPU_K8s/blob/main/FabricPortal_tests/kubernetes_simple.ipynb;
- https://github.com/cissieAB/DPU_K8s/blob/main/FabricPortal_tests/kubernetes_multi-nodes.ipynb

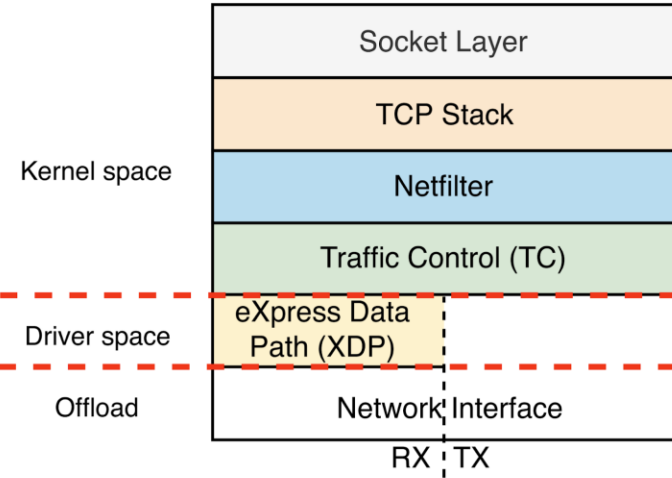
Financial Report

Will spend more hours in the future.

Budget vs. Actuals - LD2513 (\$K loaded)
DPU HPDF-H



Backup - TC



TC (traffic control) vs XDP

Thoughts & Challenges

- TC is the right way to go if we need the outbound traffic statistics.
- `sudo` permission is required every where, which raised security issues.
- Real overhead measurement and evaluation.

Polaris: Unveiling HPC Secrets with Telemetry Analysis and Beyond

Yiyang Lu, Jie Ren, Evgenia Smirni
College of William and Mary

Motivation

Unsupervised, application-agnostic analysis for the production telemetries enables system administrators and users to gain insights into system behavior, identify anomalies, and optimize resource allocation and system operation.

Polaris

- Propose a unified Fundamental Model (FM) for HPC telemetry forecasting
- Generate Telemetry Relationship Graphs (TRGs)
- Enable a wide range of telemetry analysis tasks, such as telemetry forecasting, telemetry Relationship mining, workload characterization, and anomaly detection.

Fundamental model (FM) architecture

The FM takes N telemetry variables as input with a sliding window size of T , time steps and predicts their values for the next T time steps. The model consists of L interleaved time-mixing and feature-mixing layers, each containing fully connected (FC) layers followed by activation functions and dropout.

Telemetry Relationship Graph (TRG) Construction:

```

Algorithm 1 TRG Construction
1 Input: Telemetry data  $X \in \mathbb{R}^{T \times N}$ , Fundamental Model (FM)  $f$ , perturbation magnitude  $\epsilon$ 
2 Output: Directed graph  $G = (V, E)$  with edge values  $A^{(t)}$ 
3 Initialize an empty directed graph  $G = (V, E)$  with  $N$  nodes
4 for  $i = 1$  to  $N$  do
5    $\mathcal{P} = \epsilon \times \mathcal{R}(0,1)^N$ 
6    $\tilde{X}_i = X_{:,i} + \mathcal{P}$ 
7   for  $j = 1$  to  $N$  do
8     if  $i \neq j$  then
9        $\tilde{X}_i^{(t)} = f(\tilde{X}_i^{(t-1)})$ 
10       $\tilde{X}_j^{(t)} = f(\tilde{X}_j^{(t-1)})$ 
11       $G = A^{(t)}$ 
12    end if
13  end for
14 end for
15 return Directed graph  $G$  with edge values  $A^{(t)}$ 
    
```

Evaluation

Compare Polaris's performance to GNN and VAE on the OLCF and JLab datasets.

Effectiveness of FM

Evaluating multivariate time series forecasting accuracy, training and inference time.

Model	JLab					OLCF				
	Params	RMSE	Inference Time	Training Time	Training Time	Params	RMSE	Inference Time	Training Time	Training Time
GAT	35K	8.80%	334 μ s	8.5	35K	4.69%	103 μ s	14.8	8.5	14.8
GCN	32K	6.47%	371 μ s	9.5	29K	4.14%	123 μ s	15.8	9.5	15.8
Time-Mixing-only	34K	4.69%	72 μ s	3.5	34K	4.65%	39 μ s	5.8	3.5	5.8
FM	52K	0.19%	96 μ s	5.8	22K	2.67%	41 μ s	6.8	5.8	6.8

Observation: FM achieves superior model quality, as evidenced by the lowest RMSE value, while maintaining rapid training and inference speeds compared to other models.

Effectiveness of TRGs

Polaris TRG does not exhibit any connections between constant and non-constant.

Observation: Polaris constructs high-quality TRGs across two HPC systems.

Telemetry Relationship Mining

Polaris quantifying the system ever-evolving process through TRG similarities, defined by the Jaccard similarity coefficient, assessing the overlap between the edge sets of the graphs.

The TRG on the JLab shows minor changes, while the TRG on the OLCF stabilizes.

Dataset	Duration(hours)					Nodes ratio				
	12	24	36	48	60	20%	40%	60%	80%	100%
JLab (72hours)	0.85	0.73	0.77	0.86	0.75	1	0.7	0.67	0.7	0.72
OLCF (72hours)	0.91	0.95	0.95	0.99	0.99	1	0.92	0.95	0.91	0.91

Observation: Polaris can quantify the evolution of telemetry relationships across entire HPC systems.

Workload Dynamics Characterization

Polaris extracts a TRG for each one-hour time window on every compute node in OLCF.

Co-evolution cases show job transitions cause strong variations in telemetry relationships.

Observation: Polaris provides a sequence of TRGs for each compute node to reflect system statuses under varying workloads. Among these TRGs, there is greater stability within jobs than between jobs, enabling Polaris to effectively distinguish between 'job change' and 'job persist' in the system.

Anomaly Detection

Polaris identifies anomalies using a two-step approach: 1. detect unexpected behavior in telemetry data, 2. confirm anomalies are not due to job status changes.

Evaluating anomaly detection performance:

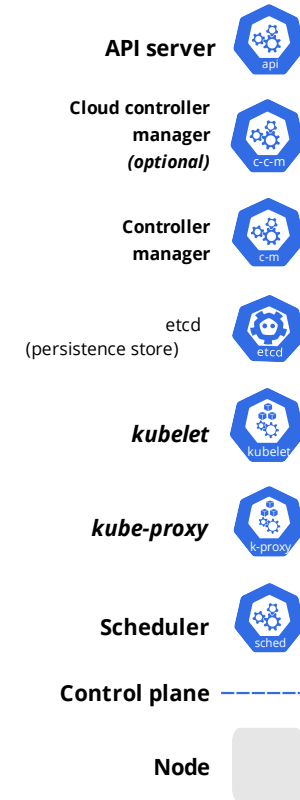
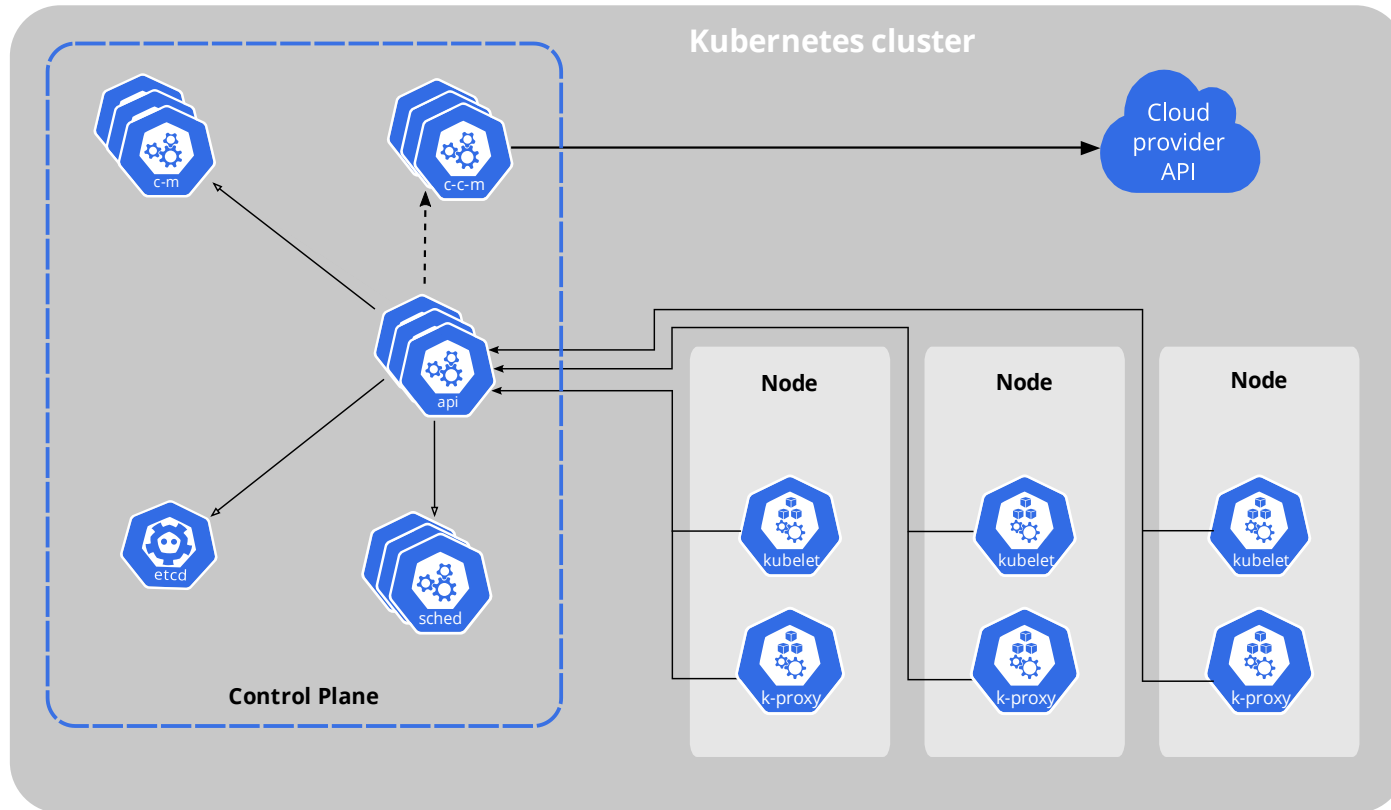
Model	JLab with synthetic anomalies				OLCF with synthetic anomalies			
	Params	F1	F2	F3	Params	F1	F2	F3
GAT	35K	0.75	0.69	0.71	35K	0.75	0.69	0.71
GCN	32K	0.74	0.68	0.70	29K	0.74	0.68	0.70
Time-Mixing-only	34K	0.82	0.80	0.81	34K	0.82	0.80	0.81
FM	52K	0.83	0.82	0.83	22K	0.83	0.82	0.83

Observation: Polaris demonstrates remarkable efficiency in anomaly detection. Step2 reduces false positives by up to 70% without sacrificing detection accuracy.

Conclusion

We present Polaris, a unified and unsupervised prediction framework for analyzing and interpreting telemetry data in HPC systems.

Backup - K8s



Thoughts & Challenges

- The behavior is not very stable on Fabric Testbed.
- Local control plane deployment on "nvidarm", with Cilium as the CNI.