Innovation and Technology Studies
Print ISSN: 3007-6919
Online ISSN: 3007-6927
DOI: https://doi.org/10.61784/its4031

AEGIS: CALIBRATED ANOMALY-RISK-DRIVEN CHECKPOINTING AND PROACTIVE MIGRATION FOR FAULT-TOLERANT HIGH-CONCURRENCY STREAM PROCESSING

XuanPing Zou[1*], KaiLang Wei[2], Zhou Lu[3]
[1]*Zoom, Inc, San Jose 95113, USA.*
[2]*Emory University, Atlanta, GA, USA.*
[3]*Northeastern University, San Jose, CA, USA.*
[*]*Corresponding Author: XuanPing Zou*

Abstract: High-concurrency cloud platforms run stateful stream-processing jobs whose availability depends on checkpoint-based rollback recovery. Production systems snapshot at static intervals tuned for an average failure rate, although cloud failures are announced by gray-failure symptoms in server telemetry and the cost of both snapshots and recovery rises sharply under bursty load. This paper presents AEGIS, a fault-tolerance controller that closes the loop between telemetry-based anomaly detection and dataflow recovery actions. AEGIS combines (i) DASS-Net, a dual-axis selective state-space detector that couples a Mamba-style causal scan over time with variate-token attention across metrics; (ii) a drift-aware conformal layer that turns residual evidence into distribution-free alarm p-values and a time-varying hazard; and (iii) a risk-integral checkpoint trigger and a break-even migration rule that generalize Young's optimal interval to time-varying hazard, state size and load. Experiments use real telemetry of 28 production servers (SMD) and the PSM fleet, and real bursty arrival traces (NAB) that drive a fault-injection study of a partitioned dataflow job with incident-correlated crashes. DASS-Net achieves the highest AUC-ROC on SMD (0.850) and raises alarms 1.5 min earlier than the most timely of 7 re-implemented baselines. Over ten held-out runs, AEGIS lowers fault-tolerance waste by 38.3% relative to Young/Daly checkpointing and by 30.8% relative to prediction-triggered migration, avoids 63.7% of crashes, reduces SLO-violating records by 55.5%, and writes 37.2% fewer snapshots.
Keywords: Cloud reliability; Fault tolerance; Stream processing; Checkpointing; Anomaly detection; State space models; Conformal prediction

1 INTRODUCTION

Stateful stream processing has become the backbone of high-concurrency cloud services such as fraud screening, real-time recommendation, observability pipelines and online feature computation. Engines in the lineage of Apache Flink execute these jobs as partitioned dataflows that ingest millions of events per second and keep large keyed state (windows, joins, aggregates) in every operator instance [1-3]. Exactly-once semantics are obtained with asynchronous barrier snapshots [4]: the job periodically persists a consistent checkpoint together with the source offsets, and after a crash it restores the last completed checkpoint and replays the input log from the recorded offsets.

The checkpoint interval embodies a delicate trade-off. Frequent snapshots consume CPU, network and storage bandwidth and interfere with record processing; infrequent snapshots inflate the amount of work that must be replayed after a failure and therefore the recovery time and the latency spike seen by downstream consumers [5]. The classical answer, the first-order optimum of Young and its refinement by Daly [6-7], prescribes a *static* interval derived from an average failure rate and an average snapshot cost. Two properties of cloud platforms invalidate both averages. First, failures are not memoryless: many fail-stop crashes are the end stage of *gray failures*, i.e., degradations that are visible in server telemetry long before the process dies [8], and production clouds already predict node failures from such signals [9-10]. Second, costs are not constant: operator state and snapshot interference grow with bursty traffic, and recovery during a burst is much slower because the backlog drains only through the spare capacity left after serving fresh arrivals. A static interval thus over-protects healthy periods and under-protects risky ones.

Prediction-aware checkpointing has been studied in high-performance computing, where failure predictors trigger proactive checkpoints or preemptive migration [11-13]. These designs assume predictors with known precision and recall and fixed alarm thresholds, whereas practical telemetry detectors emit uncalibrated scores whose distribution drifts with workload and software updates. Conversely, the multivariate time-series anomaly-detection literature optimizes offline detection metrics and rarely asks how detector outputs should drive recovery actions [14-17]. Stream-processing systems, in turn, accelerate recovery or tune intervals by profiling [5, 18-19], but treat failures as exogenous.

This paper closes this loop with **AEGIS**, an anomaly-evidence-guided fault-tolerance controller for high-concurrency stream processing. AEGIS turns raw server telemetry into a calibrated, time-varying hazard and uses that hazard, together with the live load and state size of every partition, to decide *when* to snapshot and *whether* to evacuate a partition from a suspicious host. The contributions are as follows.

(1) DASS-Net, a dual-axis selective state-space detector that couples a Mamba-style causal selective scan along time with inverted variate-token attention across metrics [20-21], and produces two complementary residuals (next-step forecast

© By the Author(s) 2024, under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0).

and bottlenecked reconstruction) for the newest observation only, which makes scoring causal and streaming-compatible. A single fleet-shared model serves all servers.

(2) A drift-aware conformal risk layer that converts residual evidence into distribution-free alarm p-values with an alarm-gated, sliding calibration bag [22], and maps alarm intensity to a fleet-fitted hazard. The false-alarm rate is set directly by the significance level instead of a dataset-specific threshold.

(3) Risk-integral decision rules: a checkpoint trigger that fires when the accrued expected rework equals the load-adjusted snapshot cost, which provably reduces to Young's interval under constant risk and cost; and a break-even migration rule with load-aware return; both rules price snapshot interference and weight the work at risk by the catch-up rate.

(4) An evaluation on real data: 28 production servers of SMD and the PSM fleet for detection [14, 23], and a fault-injection study in which real incident segments escalate into crashes while real bursty arrival traces drive a partitioned dataflow [24]. AEGIS reduces fault-tolerance waste by 38.3% and SLO-violating records by 55.5% relative to Young/Daly checkpointing, and cuts unplanned outage time by 63.7%, while taking 37.2% fewer snapshots.

2 RELATED WORK

2.1 Fault Tolerance in Distributed Stream Processing

Modern engines persist operator state with distributed snapshots and recover by rollback and replay [2, 4]; the evolution of these mechanisms is surveyed in [3]. Clonos adds causal logging and standby operators to enable fast local recovery [18], and Rhino provides handover and state-migration protocols that reconfigure queries with terabytes of state with little disruption [19]. Chiron profiles a job under different checkpoint intervals and selects the interval that satisfies quality-of-service constraints [5]. These systems treat the failure process as exogenous and stationary; AEGIS is complementary, because it decides the timing of snapshots and migrations from calibrated telemetry risk and live load, and it can drive the migration mechanisms of [19] or the recovery mechanisms of [18].

2.2 Checkpoint Optimization and Prediction-Aware Resilience

Young and Daly derive the optimal periodic interval for Poisson failures [6-7]. Bouguerra *et al.* combine failure prediction, proactive checkpoints and preventive periodic checkpoints [11], and Aupy *et al.* extend the first-order analysis to predictors characterized by recall and precision. Engelmann *et al.* migrate HPC processes away from nodes that are predicted to fail [13]. In cloud platforms, gray failures dominate incidents [8]; MING ranks failure-prone nodes from temporal and spatial signals and Narya chooses mitigation actions to avert VM interruptions [9-10]. AEGIS differs in three respects: it works with drifting score streams instead of a predictor with fixed operating point, it accounts for time-varying snapshot and recovery costs induced by dataflow load, and its rules recover the classical optimum as a special case.

2.3 Multivariate Time-Series Anomaly Detection

Reconstruction-based detectors include OmniAnomaly and USAD [14-15]; LSTM-NDT uses forecasting residuals [25]; TranAD and Anomaly Transformer adopt attention [16-17]. General forecasting backbones such as DLinear [26], PatchTST and iTransformer [21, 27], as well as selective state-space models [20], are increasingly used as detectors. Evaluation pitfalls of point-adjusted F1 are analyzed in [28], motivating event-aware metrics such as the composite F1 [29]. Isolation Forest remains a strong classical baseline [30]. DASS-Net combines the two axes that recent backbones treat separately and is designed for causal last-point scoring, which is the regime that matters for online control.

3 SYSTEM MODEL AND PROBLEM STATEMENT

We consider a keyed stream job with N partitions; partition i runs on server i, receives records at rate $r_i(t)$, serves them with capacity μ_i, and keeps state of size $S_i(t)$ that grows with the records held by its windows. A fluid model describes the backlog $B_i(t)$: records accumulate when the effective service rate falls below the arrival rate, and the queueing latency of a record arriving at t is approximately $B_i(t)/\mu_i$ plus any remaining outage. The index i is dropped when no ambiguity arises. **Snapshots.** A checkpoint started at t_b takes $C(t_b) = c_0 + S(t_b)/B_w$ seconds and reduces the service rate to $(1 - \theta)\mu$ while it runs. When it completes, the replay point moves to t_b. **Crashes.** A crash at t_f stops the partition for $D_f = D_0 + S_c/B_r$ seconds (failure detection, rescheduling and remote restore of the last snapshot of size S_c) and rewinds the source to the replay point, so that all records processed since then, the work at risk W, are reprocessed. **Migration.** A planned handover moves the partition to a healthy standby after a short pause $M(t) = m_0 + S(t)/B_m$, in the spirit of state-migration protocols [19]; the handover also resets the replay point.

Each server emits a vector of D key performance indicators (KPIs) x_t every minute. Crashes follow an unknown, time-varying hazard $\lambda(t)$; part of the hazard is announced by anomalous telemetry (escalating gray failures) and part is unpredictable. The goal is an online policy that, from telemetry and load observed up to t, chooses snapshot and migration instants to minimize the *fault-tolerance waste*

$$\Omega = \left(\theta T_{ck} + T_{down} + T_{pause} + T_{replay}\right) / T, \quad (1)$$

i.e., the fraction of service capacity lost to snapshot interference, crash downtime, migration pauses and replay, while keeping the share of records whose end-to-end latency exceeds a service-level objective (SLO) small.

4 DESIGN OF AEGIS

4.1 Overview

Figure 1 shows the control loop. Every minute, DASS-Net scores the newest telemetry vector of each server. The calibrated risk layer turns the score into a conformal p-value, an alarm intensity and a hazard estimate. Every 10 s, the controller of each partition integrates the hazard against the current work at risk, compares the result with a load-dependent snapshot cost, and decides whether to snapshot, migrate, return, or do nothing.

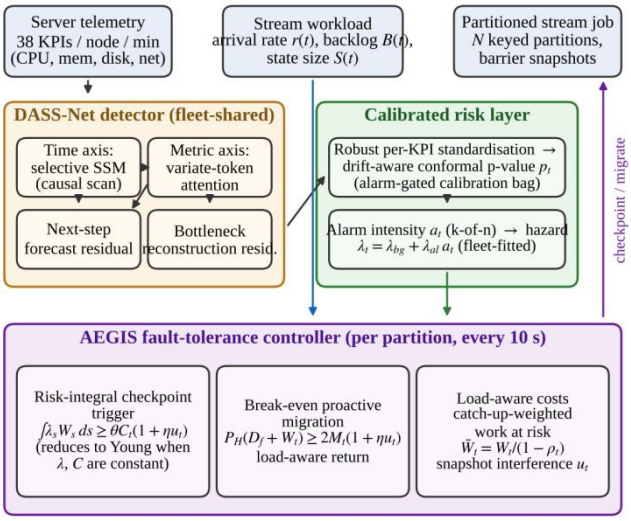

Figure 1 Architecture of AEGIS

Note: A fleet-shared DASS-Net scores every server causally; the calibrated risk layer converts residual evidence into conformal p-values and a hazard; the controller combines hazard, load and state size to trigger snapshots and migrations of the affected partition.

4.2 DASS-Net: Dual-Axis Selective State-Space Detector

Let $X_t = [x_{t-L}, \ldots, x_t] \in \mathbb{R}^{(L+1)\times D}$ be the window that ends at the newest observation. Along the **time axis**, each step is embedded, $u_\tau = E_t x_\tau$, and processed by stacked selective state-space blocks in the style of Mamba [20]. Each block applies a causal depth-wise convolution and the input-dependent recurrence

$$h_\tau = \exp(\Delta_\tau A)\, h_{\tau-1} + \Delta_\tau B_\tau u_\tau, \quad y_\tau = C_\tau h_\tau + G u_\tau, \tag{2}$$

where the step size Δ_τ and the projections B_τ, C_τ are functions of u_τ; selectivity lets the model forget quickly after a load spike and remember slowly drifting baselines. The scan is linear in L, and its state at $\tau = t-1$ depends only on past observations. Along the **metric axis**, each KPI is embedded as one variate token from its own history, $z_d = E_v x^{(d)}_{t-L:t-1}$, and the tokens interact through multi-head self-attention, which captures the cross-KPI couplings (e.g., CPU, run-queue length and network throughput) that characterize a healthy server, as in inverted Transformers [21].

Two heads read the encoders. A persistence-anchored forecaster predicts the newest observation from the past only, $\hat{x}^{(d)}_t = x^{(d)}_{t-1} + g([z_d; h_{t-1}])$, with a shared two-layer perceptron g. A bottlenecked reconstructor compresses the final temporal state through a d_{lat}-dimensional code, $\tilde{x}_t = \mathrm{Dec}(\tanh(\mathrm{Enc}(h_t)))$, which penalizes KPI combinations that are off the normal manifold even when they are predictable. The model is trained on normal data with

$$\mathcal{L} = \|\hat{x}_t - x_t\|^2 + \|\tilde{x}_t - x_t\|^2. \tag{3}$$

The per-KPI residuals $e^{f}_{t,d}$ and $e^{r}_{t,d}$ are standardized with the median and inter-quartile range of the same residuals on held-out normal data of the same server, clipped, and averaged into the score s_t. Robust standardization equalizes KPIs with very different noise levels and turns one fleet-shared network into per-server detectors, which is essential when a platform hosts thousands of machines.

4.3 Drift-Aware Conformal Risk Calibration

Raw scores are not comparable across servers or over time, so AEGIS converts them into conformal p-values. Let $\mathcal{C}_t$ be a bag of recent scores believed to be normal. Then

$$p_t = (1 + |\{c \in \mathcal{C}_t : c \geq s_t\}|) \,/\, (|\mathcal{C}_t| + 1). \tag{4}$$

When the newest normal score is exchangeable with the bag, p_t is super-uniform, so the alarm rule $z_t = 1\{p_t \leq \alpha\}$ raises false alarms with probability at most α without any labels [22]. Exchangeability is broken by drift, therefore the bag is initialized with validation scores and refreshed every hour with the latest n_{cal} scores that were *not* alarmed. The gate

prevents incidents from contaminating the reference distribution, while the sliding window tracks slow drift. Alarm intensity is a causal k-of-n average, $a_t = (1/k)\Sigma_{j<k}\, z_{t-j}$, and the hazard follows a two-state proportional model

$$\lambda_t = \lambda_{bg} + \lambda_{al} \cdot a_t, \tag{5}$$

whose parameters are the maximum-likelihood rates of crashes outside and inside alarm periods on the fleet history; λ_{bg} captures unpredictable crashes and λ_{al} the excess hazard announced by telemetry.

4.4 Risk-Integral Checkpoint Trigger

Let t_c be the last time the replay point moved and W_t the work at risk, i.e., the records processed since t_c divided by μ. Replay after a crash drains only through spare capacity, so AEGIS weights the work at risk by the catch-up factor, $\bar{W}_t = W_t/(1 - \min(\rho_t, \rho_{max}))$, where $\rho_t = r(t)/\mu$. The controller accrues the expected rework

$$A_t = \int \lambda_s \bar{W}_s \, \mathrm{d}s, \quad s \in [t_c, t], \tag{6}$$

and starts a snapshot at the first instant at which

$$A_t \geq \theta C(t) \cdot (1 + \eta u_t), \quad u_t = \mathrm{clip}\big((\varrho_t - 1 + \theta)/\theta, 0, 1\big), \tag{7}$$

where $\varrho_t = (r(t) + B(t)/60\ \mathrm{s})/\mu$ also accounts for draining the current backlog within a minute. The factor u_t measures the part of the snapshot overhead that cannot be absorbed by idle capacity and would therefore turn into backlog; η prices this interference. The rule is a break-even (rent-or-buy) decision: a snapshot is bought when the expected rework accumulated since the last one equals its price.

Proposition 1. If λ, C and ρ are constant, $u \equiv 0$ and the work at risk grows as $W_s = \rho(s - t_c)$ (no catch-up weighting), rule (7) snapshots every $\tau^* = (2\theta C/(\lambda\rho))^{1/2}$; for $\rho = 1$ this equals Young's first-order optimum with checkpoint overhead θC and mean time between failures $1/\lambda$ [6].

Proof. With constant rates, $A(\tau) = \int_0^\tau \lambda\rho s \, \mathrm{d}s = \lambda\rho\tau^2/2$; equating with θC gives the claim. ▌ Consequently AEGIS behaves like Young/Daly checkpointing when telemetry is quiet and the load is stable, shortens the interval automatically when the hazard rises or the replay becomes expensive, and defers snapshots during bursts unless the risk justifies the interference.

4.5 Break-Even Proactive Migration

A snapshot bounds the replay but not the outage itself. When the alarm intensity exceeds a_{mig}, AEGIS estimates the probability that the host crashes within a horizon H, $P_H = 1 - \exp(-(\lambda_t - \lambda_{bg})H)$, and migrates the partition if

$$P_H \cdot \big(D_f + \bar{W}_t\big) \geq 2M(t) \cdot (1 + \eta u_t), \tag{8}$$

i.e., when the expected avoided loss exceeds the pause of a round trip (evacuation and later return). The partition returns once the host has stayed alarm-free for T_{ret} minutes and the load leaves enough headroom ($u_t = 0$), so that the return pause is absorbed. Snapshots and migrations are coupled: a migration resets the replay point, and after an evacuation the partition is exposed only to the background hazard λ_{bg}, which lengthens the snapshot interval automatically.

4.6 Complexity

DASS-Net costs $O(L \cdot d \cdot N_s)$ for the scan and $O(D^2 d)$ for metric attention per scored minute and server, where N_s is the state size. The calibration layer needs one binary search in a sorted bag per score and one sort per hour. The controller performs $O(1)$ work per partition and decision step, so the control plane scales linearly with the number of partitions.

5 EXPERIMENTAL SETUP

5.1 Datasets

SMD contains 38 KPIs sampled every minute on 28 servers of a large Internet company (708,405 training and 708,420 test points; 4.16% anomalous) [14]. **PSM** contains 25 KPIs of eBay server machines (132,481 training and 87,841 test points; 27.76% anomalous) [23]. **NAB realTweets** provides real Twitter mention volumes at 5-min resolution [24]; the five series with the highest volume serve as bursty arrival-rate traces. All datasets are public and were collected before 2024.

5.2 Detection Protocol

All detectors receive the window $x_{t-32:t}$ and score only x_t, which is causal and identical to online operation. One fleet-shared model per dataset is trained for 2000 steps (batch 64, AdamW, cosine learning-rate decay) on the first 80% of the training split; the remaining 20% provides the residual statistics for standardization and the initial calibration bag. Baselines are IForest [30], LSTM-NDT [25], USAD [15], TranAD [16], DLinear [26], PatchTST and iTransformer [21, 27], re-implemented in one JAX code base with the same window, budget and residual standardization. DASS-Net uses $d = 32$, two selective-scan blocks (state size 8, convolution width 4), one four-head attention layer over variate tokens and $d_{lat} = 8$ (31k parameters). We report AUC-ROC, AUC-PR, best point-wise F1, the composite F1 (Fc1) [29], point-adjusted F1 (PA-F1, for comparability only) [28], and the mean detection delay at the Fc1-optimal threshold; SMD metrics are averaged over servers.

5.3 Fault-Injection Study

Each of the 28 SMD servers hosts one partition. Every labeled incident segment escalates into a fail-stop crash with probability $p_{esc} = 0.5$ at a uniformly random instant inside the segment, and memoryless background crashes occur at 0.1 per day and node. The first 30% of every test series is treated as operational history (hazard fitting and calibration warm-up) and the remaining 70% as the evaluation horizon (344 node-days). Arrival traces are rescaled to a median of 20,000 records/s per partition with square-root burst compression, and each partition is provisioned so that its 99.5th-percentile arrival rate uses 90% of capacity. Table 1 lists the remaining parameters. We compare Fixed-1min and Fixed-10min intervals, Young/Daly with the fleet failure rate [6-7], Dyn-Young (interval re-estimated from each node's own failure history and current snapshot cost), Pred-Ckpt (Young/Daly plus a proactive snapshot on every alarm of a static 99th-percentile threshold) and Pred-Mig (Young/Daly plus migration when the static alarm persists for 3 of 5 minutes) [10-11, 13]. All predictive policies use DASS-Net scores unless stated otherwise. The interference price $\eta \in \{2, 4\}$, the catch-up weighting (on/off) and an optional recovery-time guard of AEGIS are selected on development seeds 0–2 with a pre-declared rule (minimum SLO-violation rate among settings within 10% of the minimum waste); the selected setting is $\eta = 4$ with catch-up weighting and without the guard. All reported results use ten held-out seeds (10–19).

Table 1 Simulation Parameters

Parameter	Value
Time step / decision period	10 s
Median arrival rate per partition	20,000 records/s
State size S(t)	0.5 GB + 64 B × records in 10-min window
Snapshot: c_0, B_w, overhead θ	2 s, 200 MB/s, 0.5
Crash: D_0, restore bandwidth B_r	45 s, 100 MB/s
Migration: m_0, transfer bandwidth B_m	3 s, 1 GB/s
Conformal level α, bag size, k	0.01, 3 days, 5 min
Horizon H, a_{mig}, T_{ret}	30 min, 0.6, 30 min
SLO on end-to-end latency	5 s

6 RESULTS

6.1 Detection Accuracy and Timeliness

Tables 2 and 3 report detection quality under causal last-point scoring. On SMD, DASS-Net achieves the best AUC-ROC, PA-F1 among all 8 detectors and, most importantly for online control, it has the shortest mean detection delay: alarms arrive after 18.1 min on average, versus 19.6 min for the most timely baseline (PatchTST), i.e., 7.7% earlier. On the point-wise metrics where it is not first, the gap to the best baseline is small (AUC-PR 0.396 vs. 0.414 for LSTM-NDT; F1 0.435 vs. 0.457 for DLinear). Section 6.5 relates these metrics to end-to-end resilience.

On PSM, whose anomalies are long and frequent (27.8% of test points), DASS-Net obtains the best Fc1 (0.734) and PA-F1 (0.961), i.e., it detects the largest share of incidents at a given precision, while the highest AUC-PR (0.564 vs. 0.466) is obtained by LSTM-NDT and the shortest mean delay (3.5 min vs. 29.0 min) is obtained by TranAD. USAD, which reconstructs whole windows through a fixed latent code, degrades markedly on both datasets when one network serves many servers, while forecasting-based models adapt through their input context. Point-adjusted F1 exceeds 0.9 for most models and barely separates them, which confirms the observations of [28] and motivates our emphasis on AUC-PR, Fc1 and delay.

Table 2 Detection Results on SMD (28 Servers, Mean over Servers; Delay in Minutes, Lower Is Better)

Model	ROC	PR	F1	Fc1	PA-F1	Delay
IForest [30]	0.761	0.277	0.329	0.523	0.858	51.2
LSTM-NDT [25]	0.837	**0.414**	0.453	**0.861**	0.973	28.9
USAD [15]	0.605	0.267	0.339	0.414	0.763	49.7
TranAD [16]	0.812	0.344	0.396	0.792	0.953	31.1
DLinear [26]	0.846	0.406	**0.457**	0.764	0.948	21.9
PatchTST [27]	0.824	0.376	0.414	0.803	0.970	19.6
iTransformer [21]	0.825	0.372	0.413	0.799	0.967	22.6
DASS-Net (ours)	**0.850**	0.396	0.435	0.857	**0.977**	**18.1**

Table 3 Detection Results on PSM

Model	ROC	PR	F1	Fc1	PA-F1	Delay
IForest [30]	0.688	0.471	0.488	0.604	0.915	19.1
LSTM-NDT [25]	**0.784**	**0.564**	0.599	0.645	0.929	10.3
USAD [15]	0.533	0.413	0.437	0.435	0.688	0.0†
TranAD [16]	0.762	0.511	**0.605**	0.603	0.785	**3.5**
DLinear [26]	0.605	0.377	0.448	0.625	0.931	17.7
PatchTST [27]	0.566	0.355	0.435	0.663	0.956	22.6
iTransformer [21]	0.568	0.355	0.435	0.662	0.956	22.1

Model	ROC	PR	F1	Fc1	PA-F1	Delay
DASS-Net (ours)	0.749	0.466	0.581	**0.734**	**0.961**	29.0

Note: † The Fc1-optimal threshold flags every point, so the delay is not informative.

6.2 Ablation of DASS-Net

Table 4 removes one component at a time (servers machine-1-1 to machine-1-6 of SMD, 1,500 training steps, otherwise identical settings). Removing the metric-axis attention lowers AUC-PR from 0.488 to 0.436 and lengthens the delay to 7.9 min, showing that cross-KPI consistency carries information that the temporal scan alone misses. Removing the reconstruction head costs most (AUC-PR 0.386, Fc1 0.830), because a forecaster anchored to the last value adapts to slowly developing faults, whereas the bottleneck penalizes off-manifold KPI combinations regardless of their predictability. The forecasting head contributes 0.030 AUC-PR and is responsible for the reaction to abrupt changes. The two residuals are therefore complementary.

Table 4 Ablation of DASS-Net (Six SMD Servers)

Model	ROC	PR	F1	Fc1	PA-F1	Delay
DASS-Net (full)	0.895	0.488	0.525	0.858	0.973	7.2
w/o metric-axis attention	0.856	0.436	0.491	0.842	0.966	7.9
w/o reconstruction head	0.868	0.386	0.425	0.830	0.974	9.9
w/o forecasting head	0.846	0.458	0.504	0.874	0.963	6.3

6.3 End-to-End Fault Tolerance

Table 5 summarizes ten held-out runs over 344 node-days. Static policies expose the classical dilemma: Fixed-1min keeps replay minimal (0.47 M records/day) but its 1065.3 snapshots per node-day waste 5.35% of capacity and, because they collide with bursts, push 1.71% of the records beyond the SLO; Fixed-10min and Young/Daly snapshot far less but replay up to 7.12 M records/day. Young/Daly attains the lowest waste among static policies (0.474%), as theory predicts, yet every one of its 0.470 crashes per node-day causes a full outage and catch-up.

Prediction helps only partly when it is driven by a static threshold. Pred-Ckpt reduces replay to 4.79 M records/day but its extra snapshots raise waste to 0.512%, and it cannot avoid any crash. Pred-Mig avoids part of the incident-driven crashes (0.308 per node-day) and lowers waste to 0.423%. AEGIS dominates both: it lowers waste to **0.292%** (−38.3% vs. Young/Daly, −30.8% vs. Pred-Mig), crashes to 0.171 per node-day (−63.7%), unplanned outage to 0.166 min per node-day (−63.7%), and SLO-violating records to 0.189% (−55.5%), while taking 37.2% fewer snapshots and writing 35.4% less checkpoint data than Young/Daly. The 4.03 planned migrations per node-day cost only seconds each, because they move state over the network instead of restoring it from remote storage after a detection timeout.

Table 5 Fault-Tolerance Results (Mean of 10 Held-Out Runs; per Node-Day)

Policy	W	SLO	UO	Cr	Ck	Mg	Rp	p99.9
Fixed-1min	5.349	1.712	0.456	0.470	1065.3	0.00	**0.47**	226
Fixed-10min	0.788	0.356	0.456	0.470	138.8	0.00	3.15	**111**
Young/Daly	0.474	0.424	0.456	0.470	60.3	0.00	7.12	362
Dyn-Young	0.479	0.424	0.456	0.470	60.3	0.00	7.61	480
Pred-Ckpt	0.512	0.362	0.456	0.470	77.2	0.00	4.79	251
Pred-Mig	0.423	0.301	0.298	0.308	59.1	2.56	4.78	257
AEGIS	**0.292**	**0.189**	**0.166**	**0.171**	**37.8**	4.03	3.76	125
Oracle risk	0.214	0.184	0.111	0.115	22.8	1.40	4.52	312

Note: W: waste %, SLO: records > 5 s (%), UO: unplanned outage (min), Cr: crashes, Ck: snapshots, Mg: migrations, Rp: replayed records (M), p99.9 latency (s).

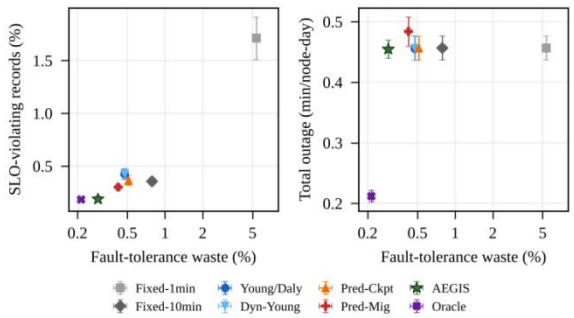

Figure 2 Trade-Offs between Fault-Tolerance Waste and (Left) SLO-Violating Records and (Right) Total Outage per Node-Day (Mean ± Std over Ten Held-Out Runs; Log-Scaled Waste Axis)

Figure 2 places all policies in the waste–SLO and waste–outage planes. In the waste–SLO plane, AEGIS is the realizable policy closest to the origin. In the outage plane, it keeps the total outage at the Young/Daly level (0.454 vs. 0.456 min per

node-day) while spending far less capacity on fault tolerance, because short planned pauses replace most crash outages; unplanned outages, which trigger replay, catch-up and paging, drop by 63.7%. The oracle row of Table 5, which receives the true incident labels instead of DASS-Net alarms, bounds what better detection could add: 0.214% waste and 0.115 crashes per node-day. AEGIS closes 69.7% of the waste gap between Young/Daly and this oracle.
Figure 3 shows the latency distribution of all records. Frequent fixed snapshots shift the whole tail because they repeatedly throttle bursts, whereas Young/Daly and Pred-Ckpt have long tails produced by replay after crashes. AEGIS lowers the probability of exceeding the SLO and shortens the extreme tail (p99.9 of 125.4 s versus 362.1 s for Young/Daly and 257.1 s for Pred-Mig); its mean latency is 0.47 s versus 1.02 s for Young/Daly.

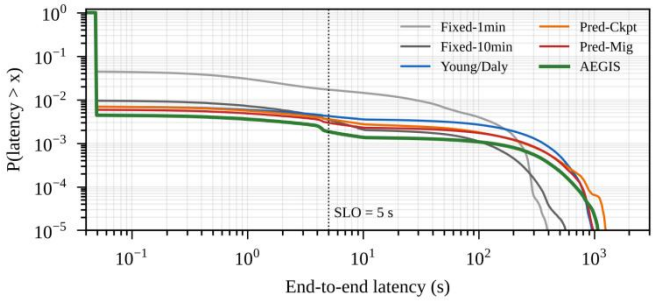

Figure 3 Complementary CDF of End-to-End Record Latency (Ten Held-Out Runs, All Partitions)
Note: The dotted line marks the 5 s SLO.

6.4 Ablation of AEGIS

Table 6 isolates each mechanism. **Calibration** is the largest single factor: replacing the conformal p-values by a static 99th-percentile threshold leaves the detector unchanged but nearly doubles the crash rate (0.309 vs. 0.171) and raises waste to 0.355%, because a threshold fixed on the validation period raises fewer persistent alarms once the score distribution drifts (2.36 vs. 4.03 migrations per node-day). **Migration** converts early warnings into avoided outages; without it, AEGIS still shortens intervals around alarms but every escalation crashes, and the waste rises to 0.486%. The **interference price** η reduces SLO violations from 0.257% to 0.189% by moving snapshots out of bursts. **Catch-up weighting** costs 0.029 percentage points of waste but cuts the p99.9 latency from 347.5 s to 125.4 s, since replay is cheap at low load and expensive near saturation. Without **telemetry risk** the rule degenerates into a load-aware Young interval (0.444% waste). Finally, driving the same calibrated alarms through fixed threshold rules instead of the risk-integral and break-even rules yields comparable SLO violations (0.188% vs. 0.189%) but 0.371% instead of 0.292% waste and 20.4 more snapshots per node-day, which quantifies the value of the decision rules themselves.

Table 6 Ablation of AEGIS (10 Held-Out Runs, per Node-Day)

Variant	Waste	SLO	Crashes	Ckpts	Migr.	p99.9
AEGIS (full)	0.292	0.189	0.171	37.8	4.03	125
w/o conformal calibration	0.355	0.309	0.309	41.5	2.36	329
w/o migration rule	0.486	0.398	0.470	66.2	0.00	264
w/o interference price ($\eta = 0$)	0.307	0.257	0.172	40.7	4.05	105
w/o catch-up weighting	0.263	0.191	0.175	29.1	3.92	348
w/o telemetry risk	0.444	0.497	0.470	43.6	0.00	524
threshold rules on calibrated alarms	0.371	0.188	0.168	58.2	4.60	50

6.5 Influence of the Detector

Table 7 plugs different detectors into AEGIS and into Pred-Mig while keeping everything else fixed. The lowest AEGIS waste is obtained with PatchTST (0.289%), and DASS-Net reaches 0.292% (a gap of 0.004 percentage points, below the run-to-run standard deviation of 0.014), while the fewest SLO-violating records are obtained with PatchTST. Across the 8 detectors, the waste achieved by AEGIS is most strongly rank-correlated with event-level Fc1 (Spearman $\rho = -0.88$) and detection delay ($\rho = 0.86$), and less strongly with point-wise F1 ($\rho = -0.64$): for proactive fault tolerance it matters whether each incident is flagged at all and early enough to act, not how many of its minutes are labeled. With identical scores, AEGIS attains lower waste than Pred-Mig for 8 of 8 detectors and fewer SLO-violating records for 7 of 8; the gain shrinks for detectors with weak ranking quality (USAD, AUC-ROC 0.605), whose calibrated alarms carry little risk information.

Table 7 AEGIS and Pred-Mig with Different Detectors (10 Held-Out Runs; Waste %, SLO %, Crashes per Node-Day)

Detector	AEGIS W	SLO	Cr	Pred-Mig W	SLO	Cr
DASS-Net	0.292	0.189	0.171	0.423	0.301	0.308
LSTM-NDT [25]	0.293	0.195	0.169	0.428	0.307	0.323
iTransformer [21]	0.294	0.185	0.172	0.419	0.296	0.297
TranAD [16]	0.313	0.229	0.216	0.414	0.284	0.290

Detector	AEGIS W	SLO	Cr	Pred-Mig W	SLO	Cr
DLinear [26]	0.304	0.225	0.201	0.424	0.302	0.312
USAD [15]	0.433	0.456	0.452	0.443	0.339	0.369
IForest [30]	0.330	0.256	0.245	0.428	0.305	0.333
PatchTST [27]	0.289	0.178	0.163	0.419	0.293	0.294

6.6 Sensitivity Analysis

Figure 4 varies the probability p_{esc} that an incident escalates into a crash. With p_{esc} = 0.25, fewer incidents escalate and the value of prediction shrinks; AEGIS nevertheless has the lowest waste (0.250% vs. 0.361% for Young/Daly and 0.322% for Pred-Mig). As p_{esc} grows to 1, the waste of static policies increases roughly linearly (Young/Daly: 0.636%), whereas AEGIS grows much more slowly (0.357%) because a larger share of crashes is announced and avoided. Doubling the snapshot cost raises the waste of Young/Daly to 0.621% and that of AEGIS to 0.367%; halving it gives 0.377% and 0.236%, respectively, so the advantage persists across a fourfold range of snapshot costs. The conformal level α trades missed alarms against unnecessary migrations: α = 0.005, 0.01, 0.02 and 0.05 give 0.321%, 0.292%, 0.267% and 0.246% waste: a larger α accepts more false alarms, which are cheap planned migrations, in exchange for earlier warnings, and every setting stays well below the waste of all baselines.

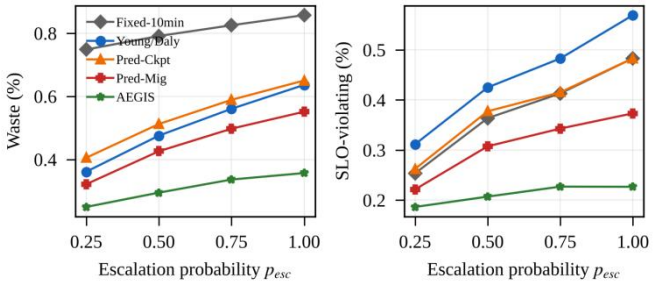

Figure 4 Sensitivity of Waste (Left) and SLO-Violating Records (Right) to the Incident Escalation Probability pesc (Five Held-Out Runs per Point)

6.7 Overhead

DASS-Net has 31k parameters. One fleet-shared model was trained for all 28 SMD servers in 338 s on a single CPU core, and scoring costs 0.56 ms per server-minute on the same core, i.e., one core can score more than 106,300 servers in real time. The controller adds a few arithmetic operations per partition and decision step, and the calibration layer one binary search per score; neither is measurable against the cost of record processing.

7 CONCLUSION

This paper presented AEGIS, a fault-tolerance controller that lets telemetry risk, live load and state size decide when a high-concurrency stream job takes snapshots and when it evacuates a partition. DASS-Net supplies early, fleet-shared anomaly evidence; drift-aware conformal calibration converts it into a hazard with controlled false-alarm rate; and risk-integral and break-even rules, which reduce to Young's interval in the stationary case, turn the hazard into actions. On real server telemetry and real bursty traffic, AEGIS reduces fault-tolerance waste by 38.3%, crashes by 63.7% and SLO-violating records by 55.5% relative to Young/Daly checkpointing. Future work will integrate AEGIS with the handover protocols of production engines, extend the hazard model to correlated failures of racks and network domains, and learn the interference price online from observed backpressure.

COMPETING INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

REFERENCES

[1] Carbone P, Katsifodimos A, Ewen S, et al. Apache Flink: Stream and batch processing in a single engine. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2015, 38(4): 28-38.
[2] Carbone P, Ewen S, Fóra G, et al. State management in Apache Flink: Consistent stateful distributed stream processing. Proceedings of the VLDB Endowment, 2017, 10(12): 1718-1729.
[3] Fragkoulis M, Carbone P, Kalavri V, et al. A survey on the evolution of stream processing systems. The VLDB Journal, 2024, 33(2): 507-541.
[4] Carbone P, Fóra G, Ewen S, et al. Lightweight asynchronous snapshots for distributed dataflows. arXiv:1506.08603, 2015.
[5] Geldenhuys M K, Thamsen L, Kao O. Chiron: Optimizing fault tolerance in QoS-aware distributed stream processing jobs. In: Proceedings of the IEEE International Conference on Big Data, 2020: 434-440.

[6] Young J W. A first order approximation to the optimum checkpoint interval. Communications of the ACM, 1974, 17(9): 530-531.
[7] Daly J T. A higher order estimate of the optimum checkpoint interval for restart dumps. Future Generation Computer Systems, 2006, 22(3): 303-312.
[8] Huang P, Guo C, Zhou L, et al. Gray failure: The Achilles' heel of cloud-scale systems. In: Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS), 2017: 150-155.
[9] Lin Q, Hsieh K, Dang Y, et al. Predicting node failure in cloud service systems. In: Proceedings of the ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE), 2018: 480-490.
[10] Levy S, Yao R, Wu Y, et al. Predictive and adaptive failure mitigation to avert production cloud VM interruptions. In: Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2020: 1155-1170.
[11] Bouguerra M S, Gainaru A, Bautista-Gomez L, et al. Improving the computing efficiency of HPC systems using a combination of proactive and preventive checkpointing. In: Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS), 2013: 501-512.
[12] Aupy G, Robert Y, Vivien F, et al. Checkpointing algorithms and fault prediction. Journal of Parallel and Distributed Computing, 2014, 74(2): 2048-2064.
[13] Engelmann C, Vallée G R, Naughton T, et al. Proactive fault tolerance using preemptive migration. In: Proceedings of the Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2009: 252-257.
[14] Su Y, Zhao Y, Niu C, et al. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2019: 2828-2837.
[15] Audibert J, Michiardi P, Guyard F, et al. USAD: Unsupervised anomaly detection on multivariate time series. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2020: 3395-3404.
[16] Tuli S, Casale G, Jennings N R. TranAD: Deep transformer networks for anomaly detection in multivariate time series data. Proceedings of the VLDB Endowment, 2022, 15(6): 1201-1214.
[17] Xu J, Wu H, Wang J, et al. Anomaly Transformer: Time series anomaly detection with association discrepancy. In: Proceedings of the International Conference on Learning Representations (ICLR), 2022.
[18] Silvestre P F, Fragkoulis M, Spinellis D, et al. Clonos: Consistent causal recovery for highly-available streaming dataflows. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD), 2021: 1637-1650.
[19] Del Monte B, Zeuch S, Rabl T, et al. Rhino: Efficient management of very large distributed state for stream processing engines. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD), 2020: 2471-2486.
[20] Gu A, Dao T. Mamba: Linear-time sequence modeling with selective state spaces. arXiv:2312.00752, 2023.
[21] Liu Y, Hu T, Zhang H, et al. iTransformer: Inverted transformers are effective for time series forecasting. In: Proceedings of the International Conference on Learning Representations (ICLR), 2024.
[22] Gibbs I, Candès E. Adaptive conformal inference under distribution shift. In: Advances in Neural Information Processing Systems (NeurIPS), 2021, 34.
[23] Abdulaal A, Liu Z, Lancewicki T. Practical approach to asynchronous multivariate time series anomaly detection and localization. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2021: 2485-2494.
[24] Lavin A, Ahmad S. Evaluating real-time anomaly detection algorithms—The Numenta anomaly benchmark. In: Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA), 2015: 38-44.
[25] Hundman K, Constantinou V, Laporte C, et al. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2018: 387-395.
[26] Zeng A, Chen M, Zhang L, et al. Are transformers effective for time series forecasting? In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2023, 37(9): 11121-11128.
[27] Nie Y, Nguyen N H, Sinthong P, et al. A time series is worth 64 words: Long-term forecasting with transformers. In: Proceedings of the International Conference on Learning Representations (ICLR), 2023.
[28] Kim S, Choi K, Choi H-S, et al. Towards a rigorous evaluation of time-series anomaly detection. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2022, 36: 7194-7201.
[29] Garg A, Zhang W, Samaran J, et al. An evaluation of anomaly detection and diagnosis in multivariate time series. IEEE Transactions on Neural Networks and Learning Systems, 2022, 33(6): 2508-2517.
[30] Liu F T, Ting K M, Zhou Z-H. Isolation forest. In: Proceedings of the IEEE International Conference on Data Mining (ICDM), 2008: 413-422.