

# Fine-grained Automated Failure Management on Extreme-Scale GPU Systems

2025 Feb 20

Balazs Gerofi <balazs.gerofi@intel.com>

Multicore World 2025, Christchurch, New Zealand



# Legal Disclaimer

**Notice: This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.**

Intel technologies may require enabled hardware, software or service activation.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Performance varies by use, configuration, and other factors. Learn more on the [Performance Index](#) site.

Your costs and results may vary.

"Conflict-free" refers to products, suppliers, supply chains, smelters, and refiners that, based on our due diligence, do not contain or source tantalum, tin, tungsten or gold (referred to as "conflict minerals" by the U.S. Securities and Exchange Commission) that directly or indirectly finance or benefit armed groups in the Democratic Republic of the Congo or adjoining countries.

All product plans and roadmaps are subject to change without notice.

Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not "commercial" names and not intended to function as trademarks.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Altering clock frequency or voltage may void any product warranties and reduce stability, security, performance, and life of the processor and other components. Check with system and component manufacturers for details.

Results have been estimated or simulated.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visiting the [Intel Resource and Document Center](#).

© 2025 Intel Corporation. Intel, the Intel logo, OpenVINO and the OpenVINO logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

# Agenda

- Motivation
- Background
  - Aurora Overview
  - Failures in Large Scale Systems
- Problem Statement
- StabilityDB Architectural Overview
- Failure Strike Policy
- Failure Management Automation
- Results
- Conclusion

# Motivation

- Failures in leadership class accelerated HPC and AI systems have become the norm rather than the exception
  - This was anticipated a decade ago in the HPC community...
- As systems continue to scale in size, the frequency of failures on the entire system is expected to increase
- Tightly coupled parallel workloads (e.g., HPC modeling/simulation and AI training/fine-tuning) are highly sensitive to failures
  - A single “*lemon node*” can ruin the run
- To ensure efficient deployment and operation, automated failure management is essential

**Original Article** This article is a U.S. government work, and is not subject to copyright in the United States.

**HIGH PERFORMANCE COMPUTING APPLICATIONS** The International Journal of High Performance Computing Applications 2014, Vol. 28(2) 129–173 DOI: 10.1177/0890059313502373 http://journals.sagepub.com/doi/10.1177/0890059313502373 http://journals.sagepub.com

**Addressing failures in exascale computing**

**Marc Snir<sup>1</sup>, Robert W. Wisniewski<sup>2</sup>, Jacob A. Abraham<sup>3</sup>, Srinivas V. Adve<sup>4</sup>, Saurabh Bagchi<sup>5</sup>, Parav Bala<sup>1</sup>, Jim Belak<sup>6</sup>, Pradip Bose<sup>7</sup>, Franck Cappello<sup>8</sup>, Bill Carlson<sup>9</sup>, Andrew A. Chien<sup>10</sup>, Paul Coteau<sup>11</sup>, Nathan A. DeBardeleben<sup>10</sup>, Pedro C. Diniz<sup>11</sup>, Christian Engelmann<sup>12</sup>, Mattan Erez<sup>13</sup>, Saverio Fazzari<sup>13</sup>, Al Geist<sup>12</sup>, Rinku Gupta<sup>1</sup>, Fred Johnson<sup>14</sup>, Sriram Krishnamoorthy<sup>15</sup>, Sven Leyffer<sup>16</sup>, Dean Liberty<sup>14</sup>, Subhashish Mitra<sup>17</sup>, Todd Munson<sup>1</sup>, Rob Schreiber<sup>18</sup>, Jon Stearley<sup>19</sup>, and Eric Van Hensbergen<sup>20</sup>**

**Abstract** We present here a report Utah, 4–11 August 2012. The levels in a computing layers of an exascale system were identified and discussed. The workshop brought from industry, government, allowed broad and complete discussion.

**Llama Team, AI@Meta**  
1 A detailed contributor list can be found in the appendix of this paper.

**Keywords** Reliability, fault-tolerance, Report on a workshop organized

**I Introduction**  
The problem was solved but by changing what we Ludwig Wittgenstein

This article is the result of failures in exascale computing. 4–11 August 2012. The we Institute for Computing is mission of the IEC is to active research and provide a way to establish a community across all the levels in a computing system. In addition to edge on resilience and hardware layers of an exascale those results, examining hardware and software, and combined approach.

**1 Introduction**  
The problem was solved but by changing what we Ludwig Wittgenstein

This article is the result of failures in exascale computing. 4–11 August 2012. The we Institute for Computing is mission of the IEC is to active research and provide a way to establish a community across all the levels in a computing system. In addition to edge on resilience and hardware layers of an exascale those results, examining hardware and software, and combined approach.

**Revisiting Reliability in Large-Scale Machine Learning Research Clusters**

Apostolos Kokkinis<sup>1</sup>, Michael Kachnik<sup>1</sup>, John Hoffmnn, Aditya Kumar, Parth Malani, Faye Ma, Zachary DeVos, Shubek Senapati, Kalyan Saladi, Carole-Jean Wu FAIR at Meta

**Abstract** Reliability is a fundamental challenge in operating large-scale machine learning (ML) infrastructures, particularly at the scale of ML models and training clusters, continue to grow. Despite significant improvements in reliability, the rate of hardware failures across different scales remains unclear. Thus, we provide quantitative analysis, operational experience, and our perspective in understanding and addressing reliability concerns in large-scale ML training clusters. We find that the most vulnerable to failures, smaller jobs make up the majority of the workloads. We also find that ML training clusters are more likely to fail than other ML environments. We identify key workload properties, compare them across clusters, and demonstrate the reliability requirements for different workloads.

**1. Introduction**  
Foundation models are to support a large variety of applications. The development of the model is trained at and G2 a post-training and improve specific capabilities. This paper provides a detailed analysis of model-native and Transformer with 40GB member of the herd is which we will refer to as We believe there are three managing components.

**• Data.** Comparing the quality of more causal and rigorous quality a corpus of about 100,000 images.

**• Scale.** We train a pre-trained model using we pre-trained a

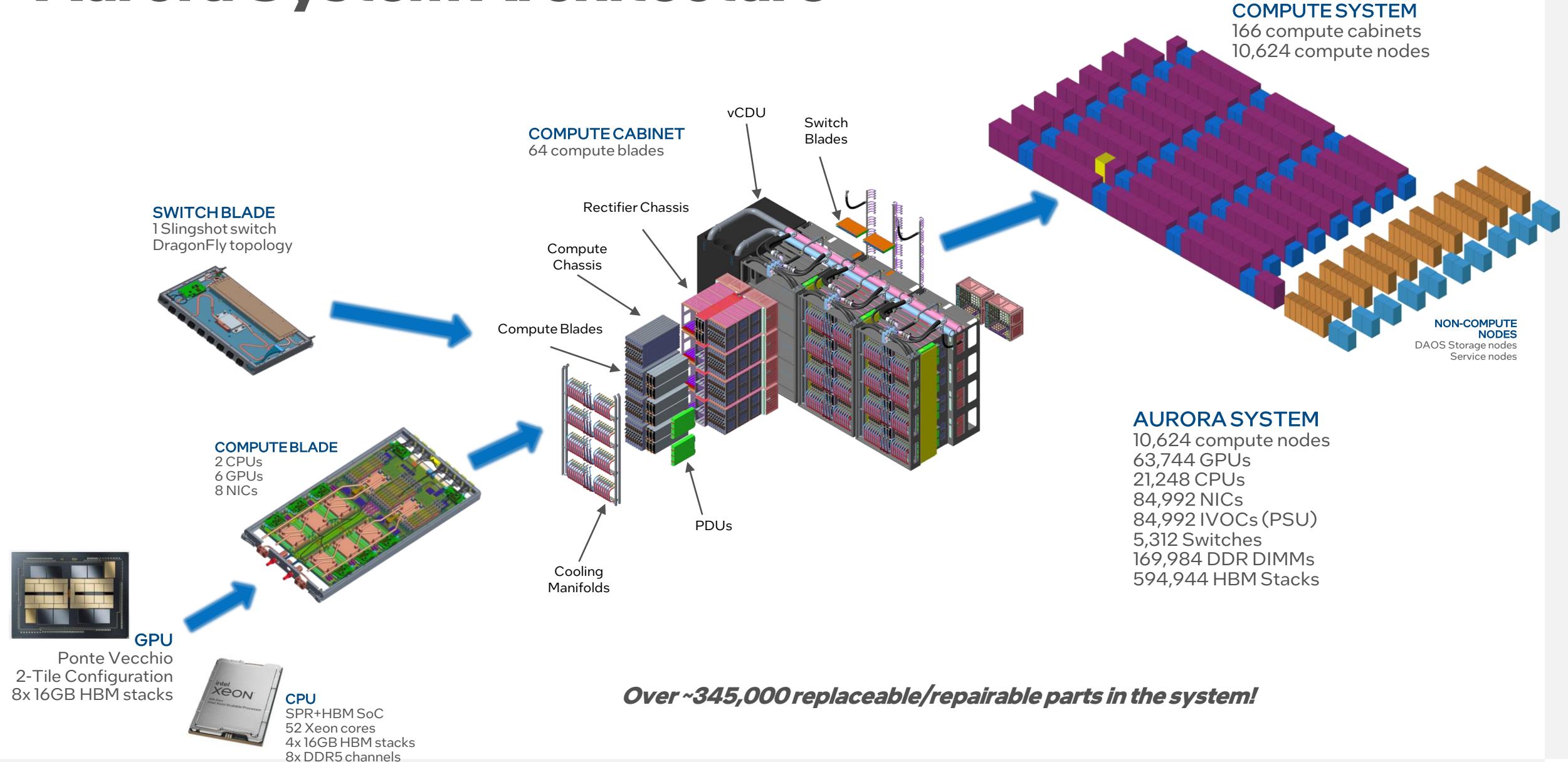
**With the rise of Large Language Models (LLMs) – Megatron [23], LLaMa [44], Gemini [12], GPT4 [35] – ML training shifted the scale of a single training job from tens to thousands of accelerators, presenting concrete challenges in reliability. While hardware failures, such as failures are not a matter of if, but a matter of when. Thus, system design entails new and interesting challenges as the operating environment of a cluster changes to support new solutions. Indeed, hardware failures [12, 13] are just some of the issues that are individually rare yet become increasingly likely as scale, involving solutions that span from the cluster level to the job level and individual accelerators.**

In this paper, we present our experience toward training a plethora of large scale models, including earlier work on Gemini [12] and LLaMa [44], and the latest work with largest jobs utilizing 48 GPUs or more. Unlike prior work, our hardware and software infrastructure is tailor-designed for ML training, and thus, we can provide a detailed analysis of ML environments with over 100 million 1.100 GPU hours and 100,000 training jobs. We also find that smaller jobs, which constitute less than 1% of our jobs while consuming 12% of the GPU resources at the cluster level. Our experience catering to both large- and small-scale jobs demonstrates the difficulty in managing failures that is rarely considered in more specialized clusters devoted to LLMs only.

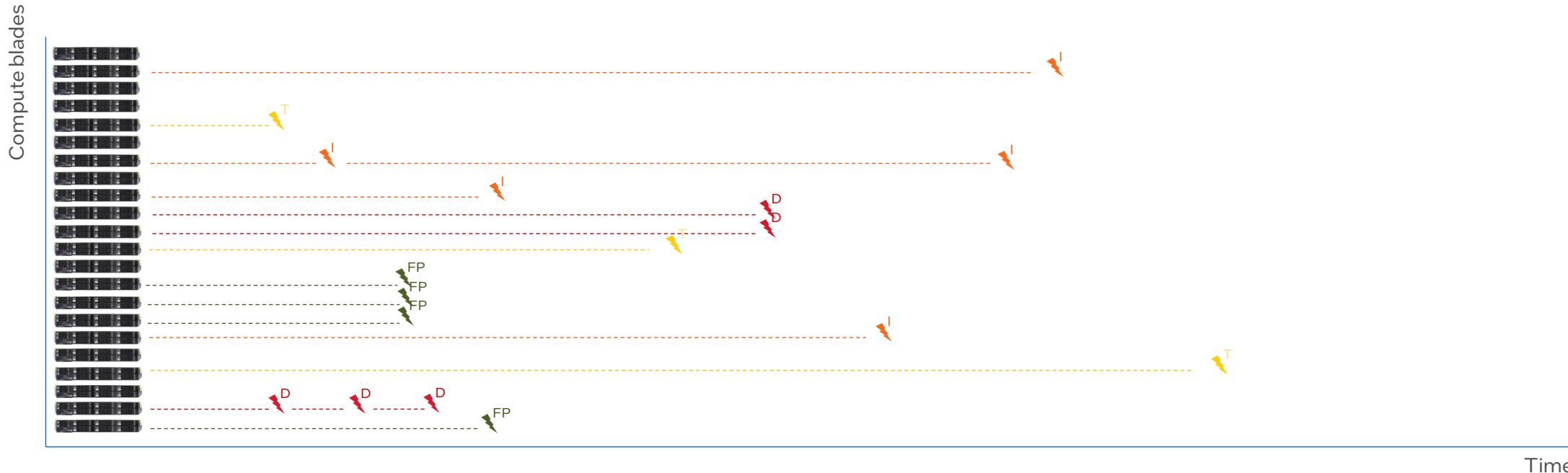
Understanding the underlying causes of job failures – let it be hardware, software, application, or some combination of the above – is key to improving training reliability, and advancing large model development. In this paper, we also present a comprehensive analysis from the perspective of AI researcher clusters with >80% utilization. The results based on real-world training systems highlight the diversity of research workloads and the challenges of maintaining reliability in such environments. Finally, we have learned how to mitigate failures at scale, tracking reliability metrics, making infrastructure changes, and diagnosing common failures. We also highlight opportunities for future research opportunities. In doing so, we provide and analyze several component failure rates, including Mean Time to Failure (MTTF).

**2. Related Work**  
To the best of our knowledge, we present the first infrastructure analysis of ML research workloads at the 10<sup>3</sup> GPU

# Aurora System Architecture



# Hardware Failures in Large Scale Systems



(D) Failures due to design bugs which fail on every instance of component <X> under specific conditions (e.g., MERT accumulator overflow bug in PVC)

(I) Failures due to intermittent faults

- Aged/degraded components, systematic issues due to marginalities (could also be deficiencies in design, validation, etc.), manufacturing defects

(T) Transient/random failures due to particle/EM radiation/cosmic rays

- These will create a “background” noise of failures that should be distributed evenly across the system

(FP) Software, networking, or correlated faults and other errors that could create failures that look like hardware component issues

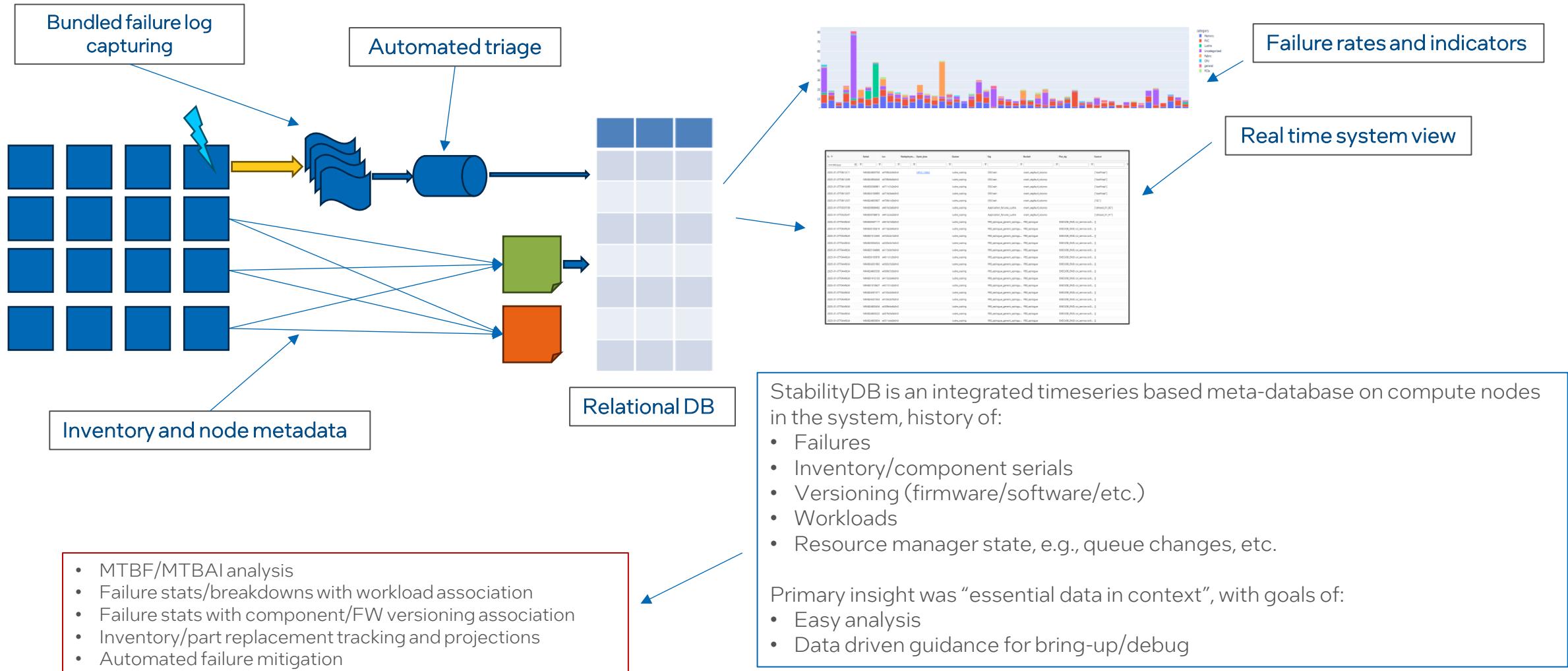
# Key Observations and Problem Statement

- How to distinguish *intermittent* and *transient* failures?
- How to handle first strikes?
  - Replacing every component on first strikes is *impractical*
- Probability of an intermittent error occurring on the same component twice is extremely small
  - Indication of defects?
- Need to understand *reoccurrence rates* and the *statistical properties of durations between strikes*
  - These are specific to failure modes
- Failure history needs to be captured *in context*
  - Firmware/software versions, external conditions have impacts
- Need for automated failure categorization
- Automated failure servicing/management?

# Agenda

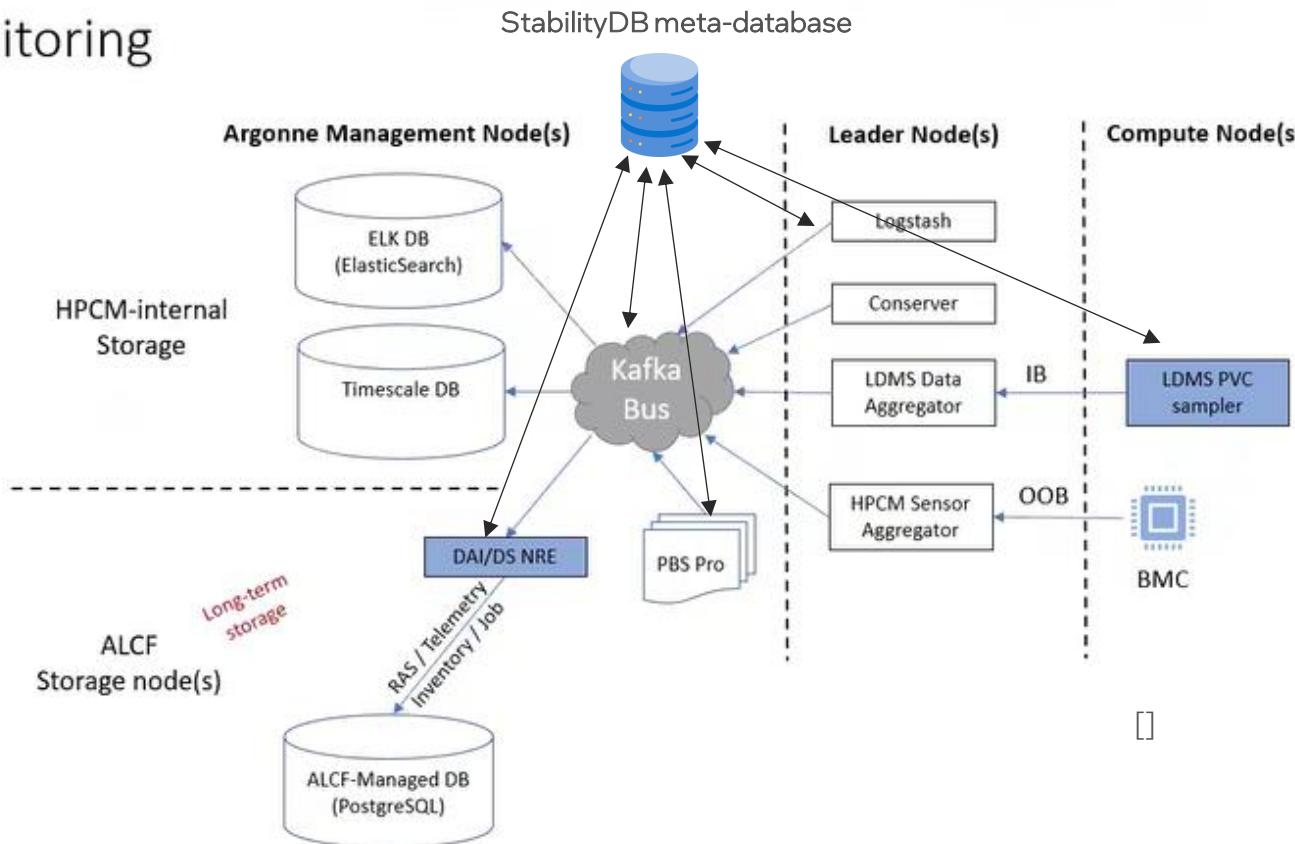
- Motivation
- Background
  - Aurora Overview
  - Failures in Large Scale Systems
- Problem Statement
- StabilityDB Architectural Overview
- Failure Strike Policy
- Failure Management Automation
- Results
- Conclusion

# StabilityDB Infrastructure Architectural Overview



# Aurora: Interaction with Cluster Management/Telemetry

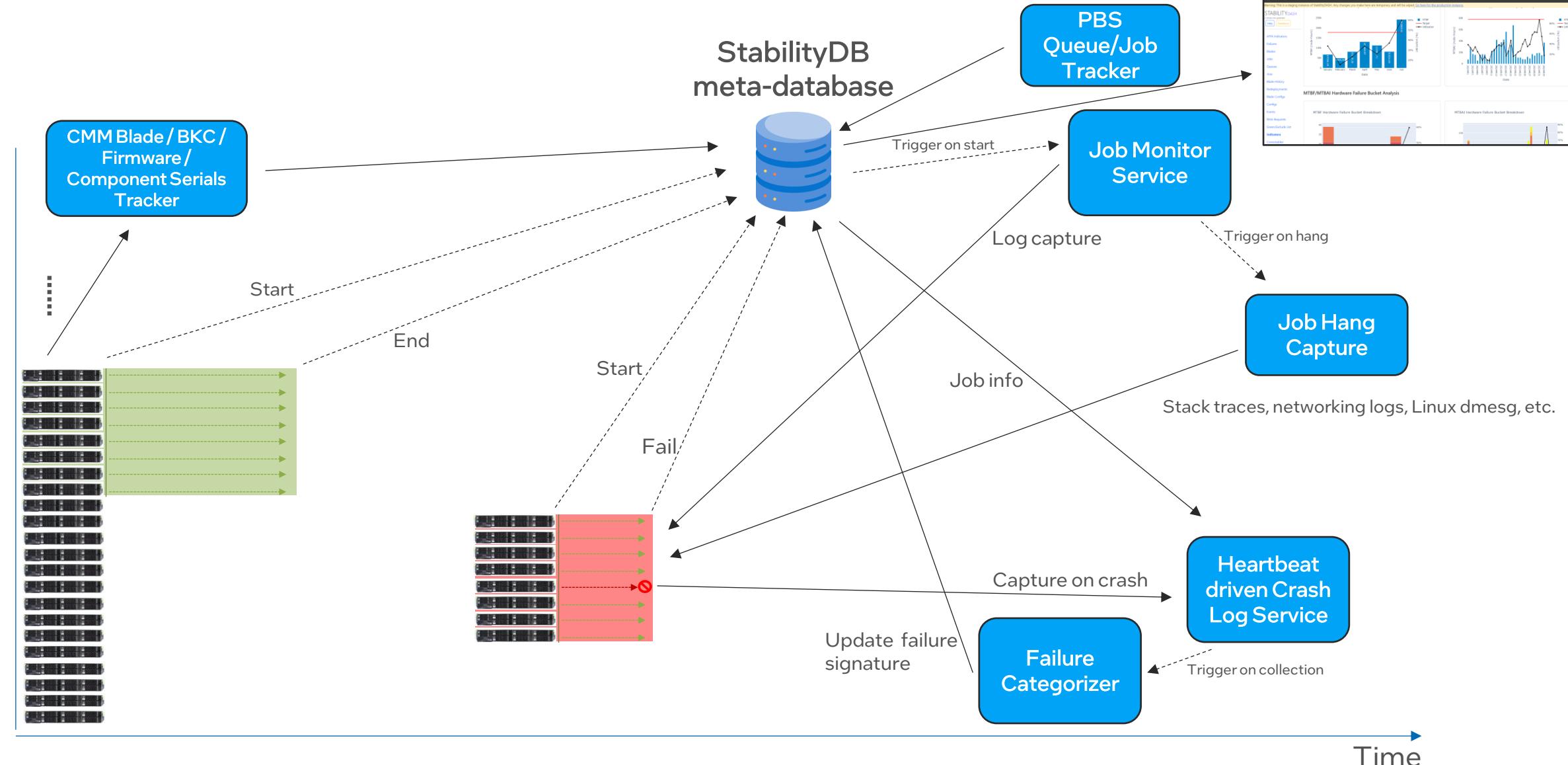
## Aurora Monitoring



- Plugs into standard telemetry/event data streams:
  - E.g., Kafka and RabbitMQ
- Interacts with cluster management software components (e.g., HPE HPCM node management)
- Interacts with batch scheduler (e.g., PBS)

# StabilityDB Backend Functional Components

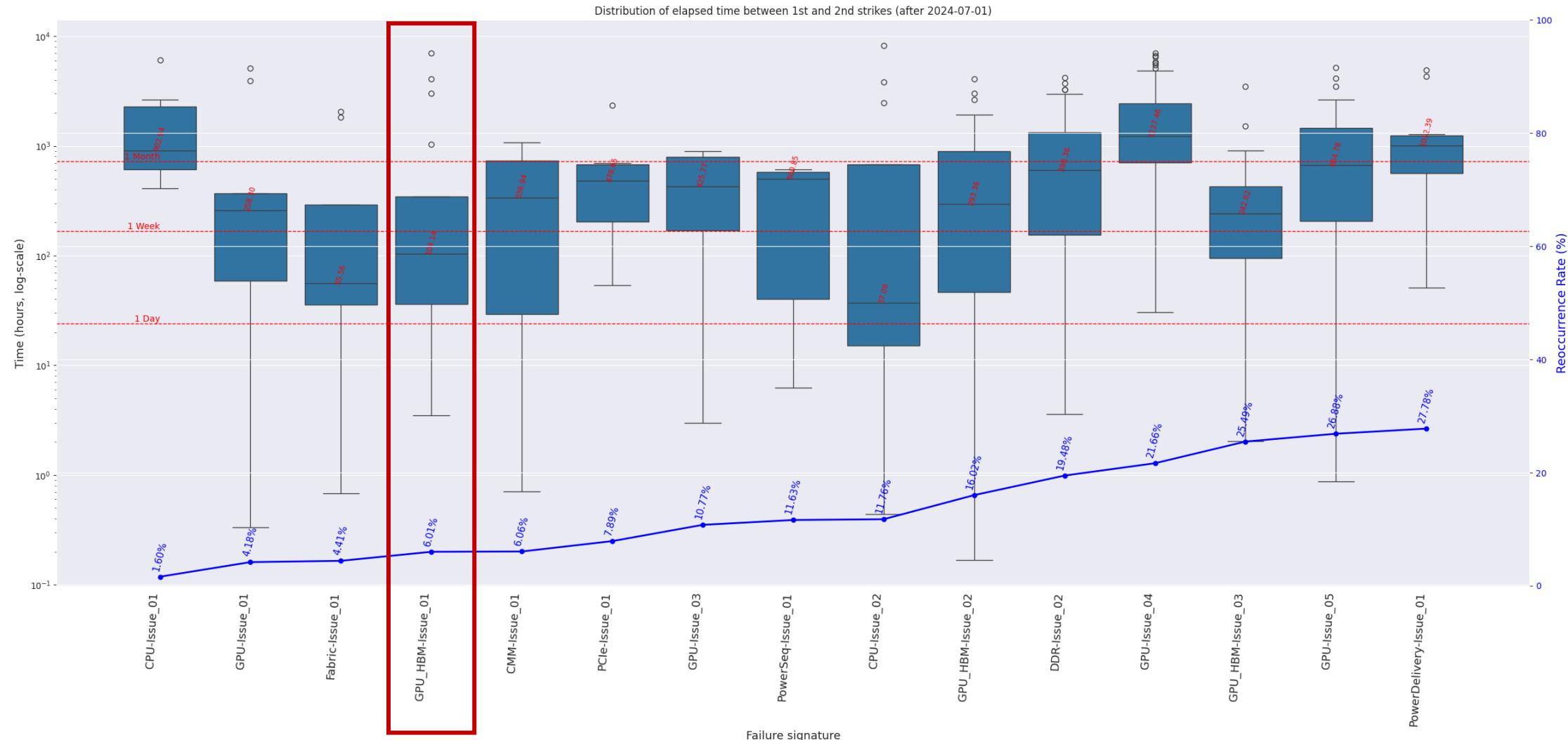
Dashboards



# Agenda

- Motivation
- Background
  - Aurora Overview
  - Failures in Large Scale Systems
- Problem Statement
- StabilityDB Architectural Overview
- Failure Strike Policy
- Failure Management Automation
- Results
- Conclusion

# Distribution of Elapsed Time between 1<sup>st</sup> and 2<sup>nd</sup> Strikes



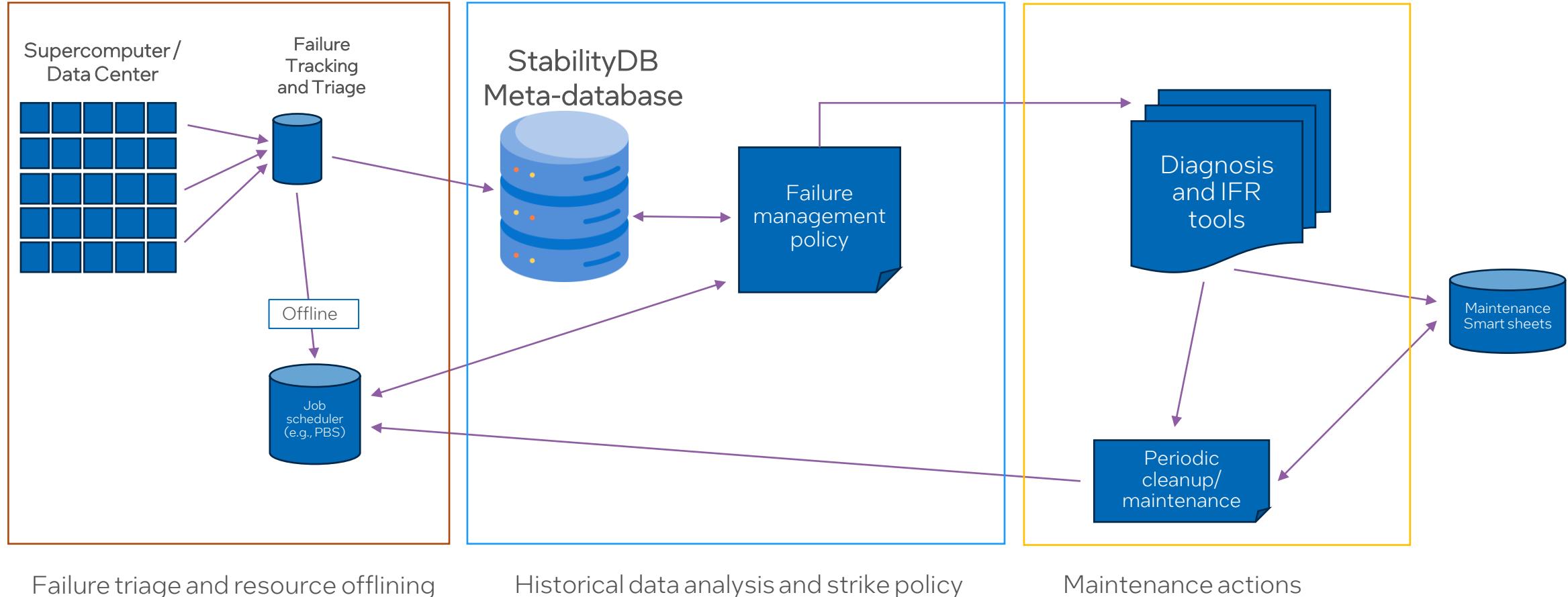
- Low reoccurrence rate
- Higher reoccurrence rate but long elapsed time between strikes

# Fine-Grained Multi-Strike Policies

Updated_ts	Sig	Live	Criteria	Strike_1	Strike_2	Strike_3	Strike_4	Strike_5
2024-12-08 13:02:41	fatal_7_multiple_quads	1	single chip	HOST_COLD_RESET,REDEPLOY_STABILITYDB... RMA_BTK		CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	fatal_7_others	1	single chip	HOST_COLD_RESET,REDEPLOY_STABILITYDB... RMA_BTK		CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	fpu_fatal_singlesubslice_error	1	single source	DSS_SWAP.MONITOR	DSS_SWAP.MONITOR	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	general_multibank_correctable_error	1	single chip	HOST_COLD_RESET,REDEPLOY_STABILITYDB... REDEPLOY_STABILITYDB.MONITOR	REDEPLOY_STABILITYDB.MONITOR	REDEPLOY_STABILITYDB.MONITOR	REDEPLOY_STABILITYDB.MONITOR	REDEPLOY_STABILITYDB.MONITOR
2024-12-08 13:02:42	general_multibank_error	0	single chip	HOST_COLD_RESET,REDEPLOY_STABILITYDB... GT IFR	RMA_BTK	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	general_multisubslice_correctable_error	1	single chip	HOST_COLD_RESET,REDEPLOY_STABILITYDB... REDEPLOY_STABILITYDB,MONITOR	REDEPLOY_STABILITYDB,MONITOR	REDEPLOY_STABILITYDB,MONITOR	REDEPLOY_STABILITYDB,MONITOR	REDEPLOY_STABILITYDB,MONITOR
2024-12-08 13:02:42	general_multisubslice_error	1	single chip	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	general_singlesubslice_error	0	single source	DSS_SWAP.MONITOR	DSS_SWAP.MONITOR	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	grf_fatal_singlesubslice_error	1	single source	DSS_SWAP.MONITOR	DSS_SWAP.MONITOR	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	hif_singlesubslice_correctable_error	1	single source	DSS_SWAP.MONITOR	DSS_SWAP.MONITOR	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	GPU_HBM-Issue_01	1	single source	HBM_IFR,PVC_COLD_RESET,MONITOR	RMA_BTK	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	hbm_controller_multi_errors	0	single chip	HBM_IFR,PVC_COLD_RESET,MONITOR	RMA_BTK	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	hbm_data_errors	1	single source	HBM_IFR,MONITOR	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP
2024-12-08 13:02:42	hbm_training_errors	1	single source	HBM_IFR,PVC_COLD_RESET,MONITOR	RMA_BTK	CONTACT_INTEL_REP	CONTACT_INTEL_REP	CONTACT_INTEL_REP

- Per failure mode fine-grained description
- Policies must be based on statistical information, for that we need a meta-database tracking those metrics
- Defines failure management automation actions

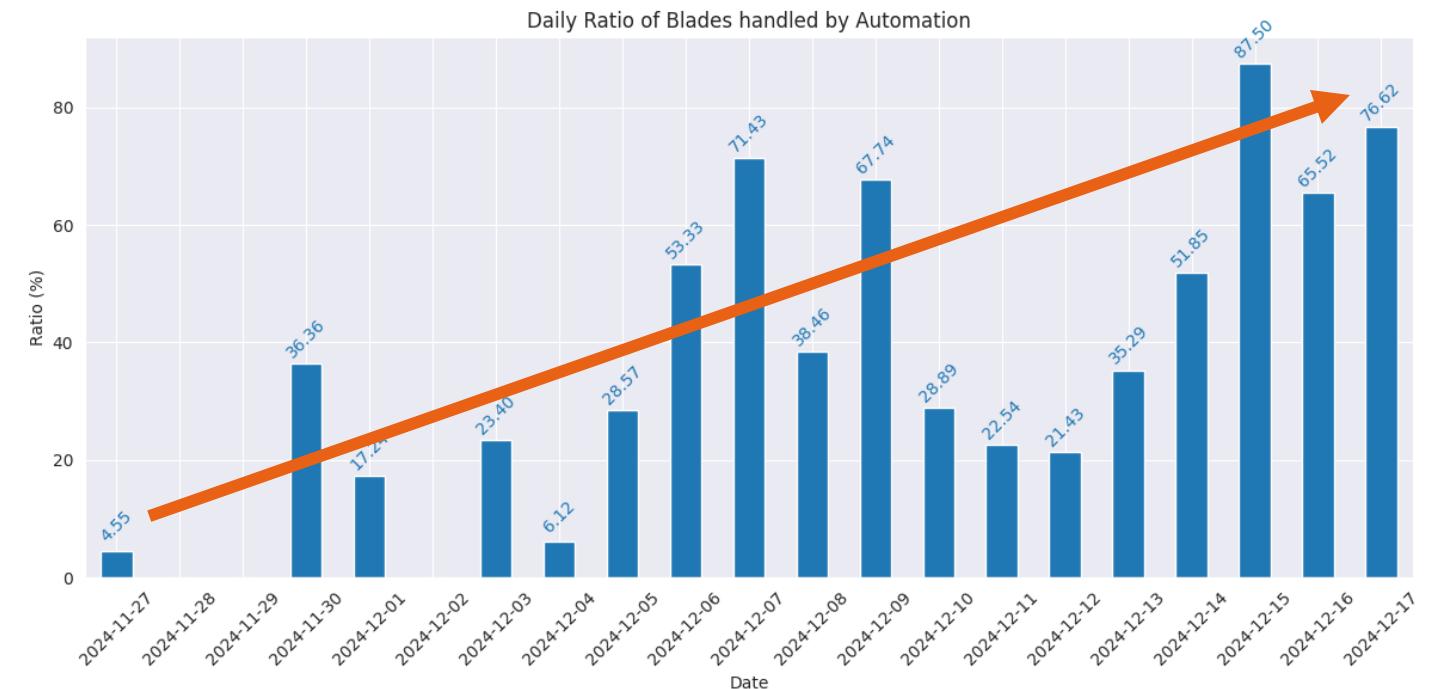
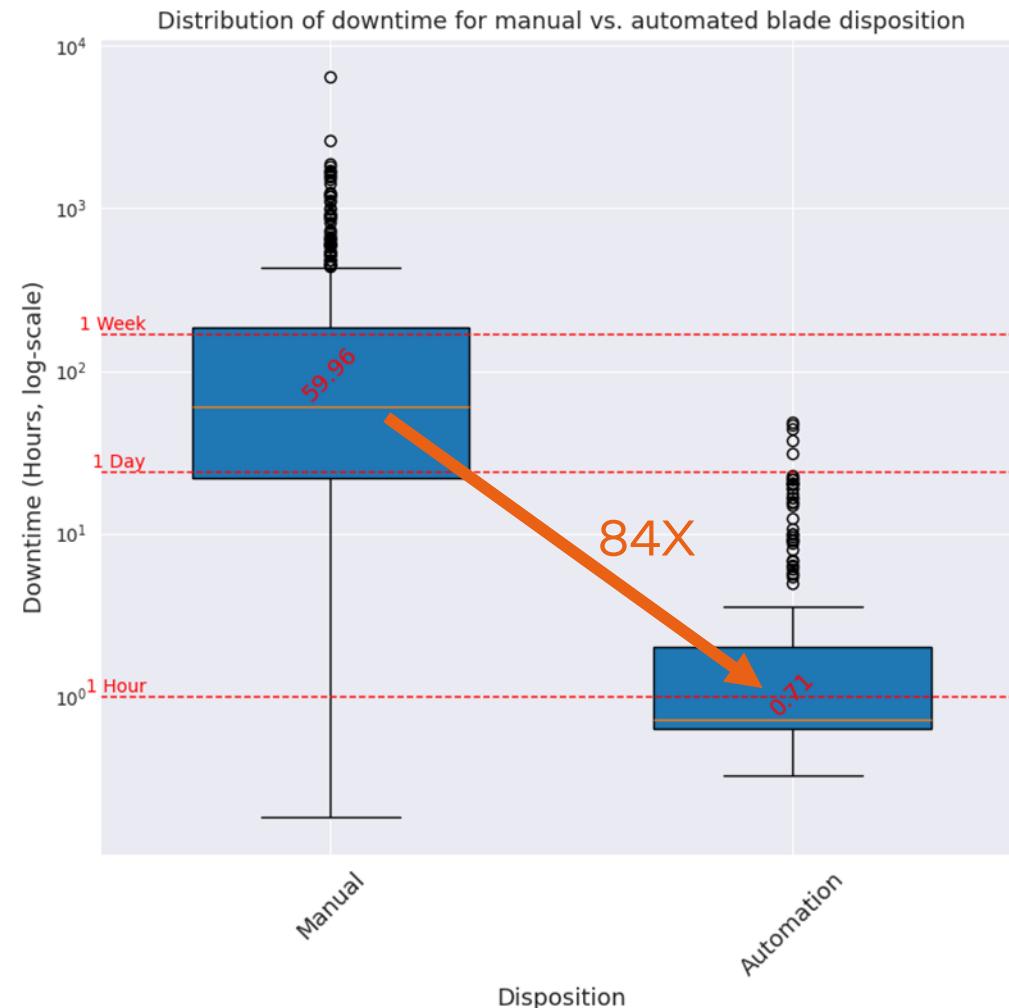
# Automated Failure Management Components



# Agenda

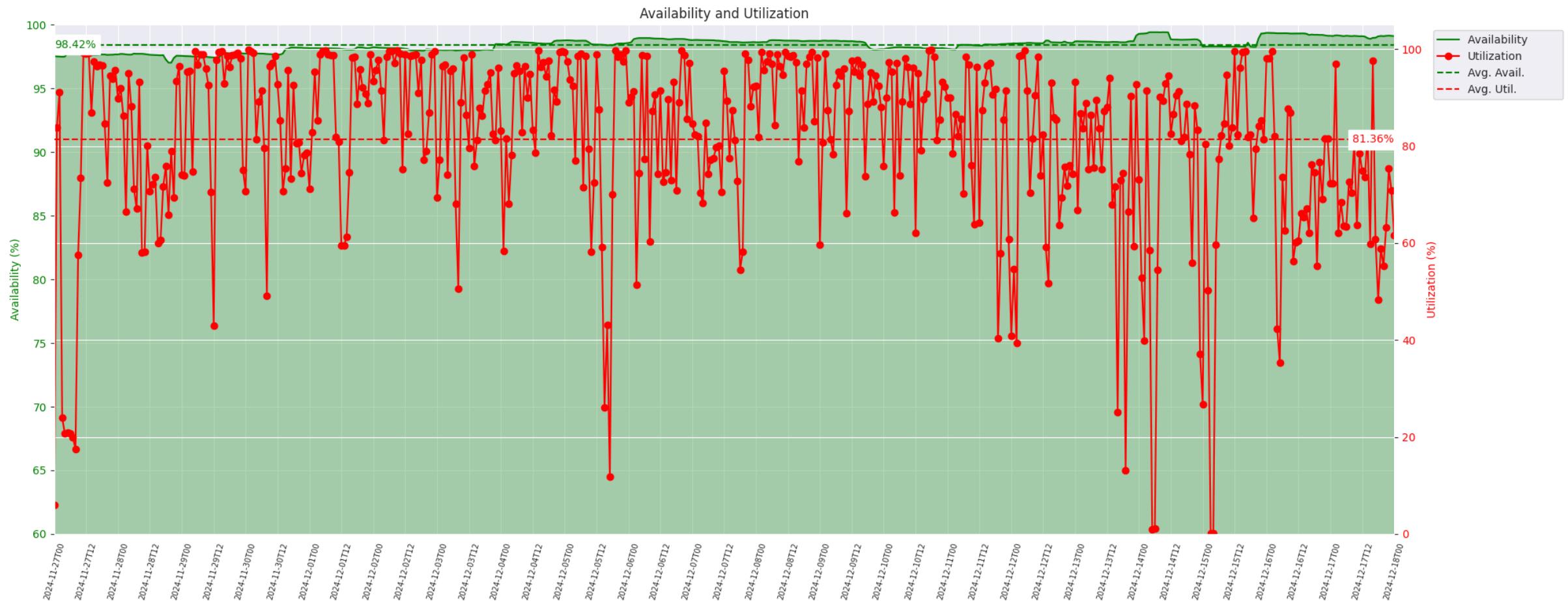
- Motivation
- Background
  - Aurora Overview
  - Failures in Large Scale Systems
- Problem Statement
- StabilityDB Architectural Overview
- Failure Strike Policy
- Failure Management Automation
- Results
- Conclusion

# Automated Failure Management Results (AT-S)



- Excludes large correlated events (e.g., Lustre mount failures, rack EPO events, etc.)
- Median time spent on addressing issues is 2 orders of magnitude lower!
- Automation's ratio steadily increasing for blade dispositions

# Availability and Utilization during AT-S



- Requirement of 95% availability in average

# First and Second Strikes Month over Month

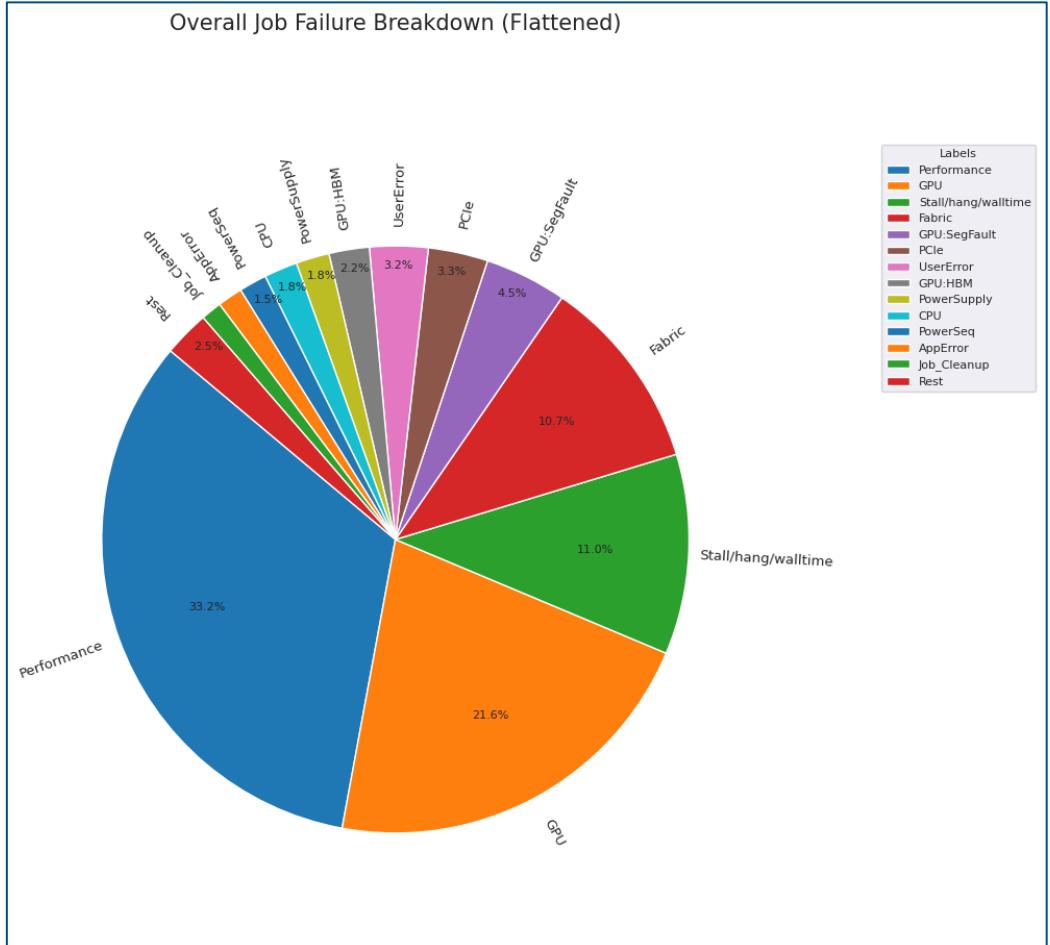
First and second strike counts per failure signature per month



- System stabilizing via decreasing 1<sup>st</sup> and 2<sup>nd</sup> counts for most failure signatures
- Constant rate of CEs on DDR and HBM

# Job Failure Breakdown during AT-S vs. Meta

\*[The Llama3 herd of models](#)



Very similar overall failure profiles!

This equals to an MTBAI = 3.7 H @ 16K GPUs  
 Scaling this linearly to Aurora scale:  
 MTBAI = 55 min @ 64K GPUs

Component	Category	Interruption Count	% of Interruptions
Faulty GPU	GPU	148	30.1%
GPU HBM3 Memory	GPU	72	17.2%
Software Bug	Dependency	54	12.9%
Network Switch/Cable	Network	35	8.4%
Host Maintenance	Unplanned Maintenance	32	7.6%
GPU SRAM Memory	GPU	19	4.5%
GPU System Processor	GPU	17	4.1%
NIC	Host	7	1.7%
NCCL Watchdog Timeouts	Unknown	7	1.7%
Silent Data Corruption	GPU	6	1.4%
GPU Thermal Interface + Sensor	GPU	6	1.4%
SSD	Host	3	0.7%
Power Supply	Host	3	0.7%
Server Chassis	Host	2	0.5%
IO Expansion Board	Host	2	0.5%
Dependency	Dependency	2	0.5%
CPU	Host	2	0.5%
System Memory	Host	2	0.5%

Table 5 Root-cause categorization of unexpected interruptions during a 54-day period of Llama 3 405B pre-training. About 78% of unexpected interruptions were attributed to confirmed or suspected hardware issues.



# Supercomputer/Data Center Digital Twins?

- “Digital twins provide living digital models of physical systems that enable **data-driven analysis** and application of artificial intelligence to better manage the datacenter and **drive efficiency for sustainability**.” [1]
- “Historically, data center management has been split into silos that each focus on one aspect.. as a result, ... different areas can miss the bigger picture. Digital twins help to **centralize data from across different areas of concern into a shared environment**” [2]
- **Areas of potential:**
  - Design: placing new servers, increasing density, improving thermal performance, etc.
  - Construction: streamlining construction, **reducing waste**, etc.
  - Operations: automating data center processes, **efficient maintenance and repairs**, etc.
  - Planning: ensuring compliance with data twins, understanding material impact, etc.

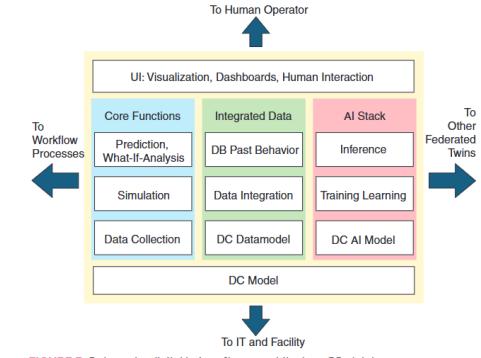
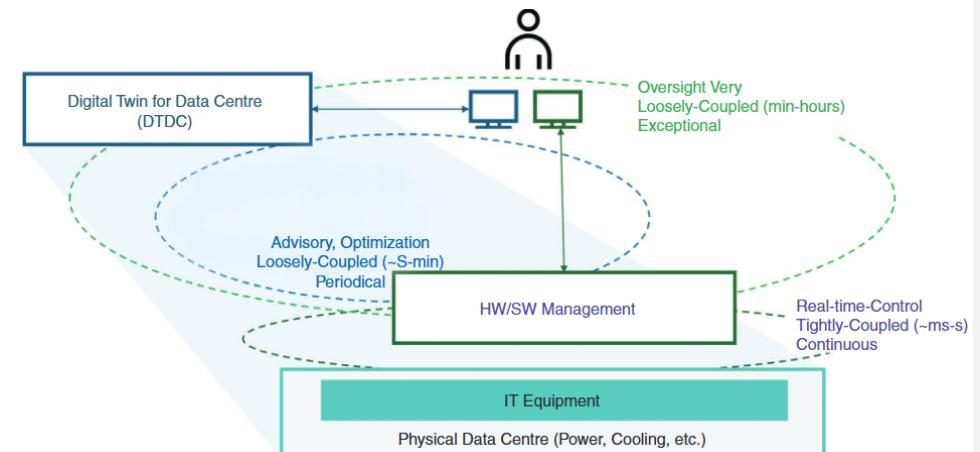


FIGURE 5. Data center digital twin software architecture. DB: database.  
<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&amumber=10687340>



<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&amumber=10687340>

[1] J. Athavale et al., "Digital Twins for Data Centers," in Computer, vol. 57, no. 10, pp. 151-158, Oct. 2024, doi: 10.1109/MC.2024.3436945.

[2] <https://venturebeat.com/ai/19-ways-digital-twins-improve-data-center-sustainability/>

# Conclusions and Outlook

- As large scale systems grow in size intermittent failures become more prevalent
- Efficient operation requires automated failure management
- Fine-grained multi-strike management policy
- Key is data in context that enables real-time decision making
- Outlook:
  - Predictive (AI?) failure avoidance
  - Continuous fleet scanning
  - Standardized failure reporting across components?

# Acknowledgments/Contributors

- Yoni Levitt
- Richard Barella
- Erik Adames
- Leobardo Rountree
- Sam Zeltner
- Spurthi Lokeshappa
- Tom Musta
- Damon Millar
- Aravind Balasubramanian
- Patrick Steinbrecher
- Arjun Kripańidhi
- Aakash Patel
- Neha Gupta
- Kevin Canada
- Ky Merril
- Brian Holland
- Sucheta Raghunanda
- Ben Allen (ANL)
- Peter Upton (ANL)
- Doug Waldron (ANL)

**Thank you for your attention!  
Questions?**

The Intel logo is displayed in white on a solid blue background. The word "intel" is written in a lowercase, sans-serif font. A small, solid blue square is positioned above the letter "i". The letter "i" has a vertical stroke on its left side. The letter "t" has a vertical stroke on its right side. The letter "e" has a vertical stroke on its left side. The letter "l" has a vertical stroke on its right side. A registered trademark symbol (®) is located at the bottom right of the letter "l".