USENIX ATC '24

# RL-Watchdog: A Fast and Predictable SSD Liveness Watchdog on Storage Systems


Jinyong Ha
Seoul National University

Sangjin Lee
Chung-Ang University

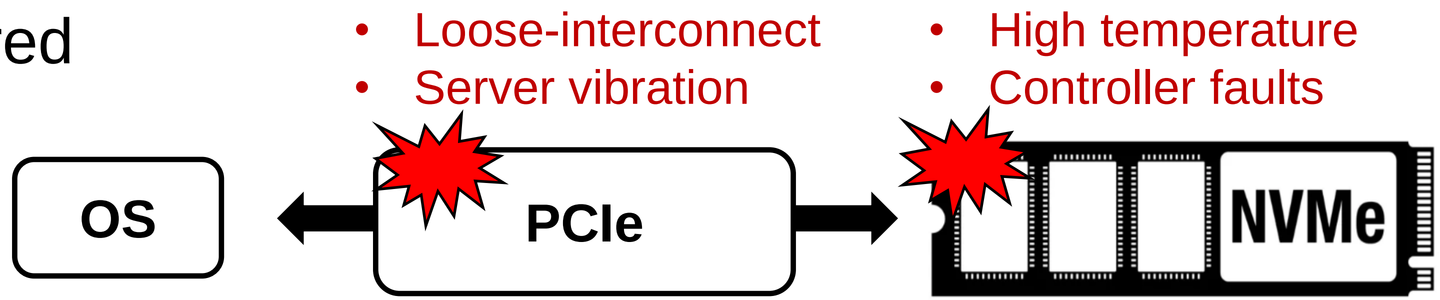
Heon Young Yeom
Seoul National University

Yongseok Son
Chung-Ang University

# Liveness of SSDs

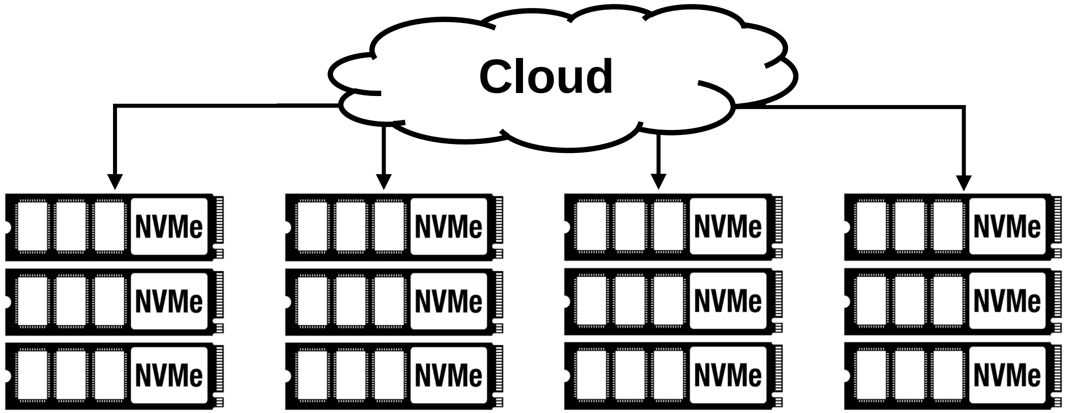
## ❖Unresponsive SSDs (SSD failure) by **faults**

- System cannot use SSDs
- Post-failure process required
  - Recovery, reboot, etc.




## ❖As more SSDs are utilized on huge storage systems

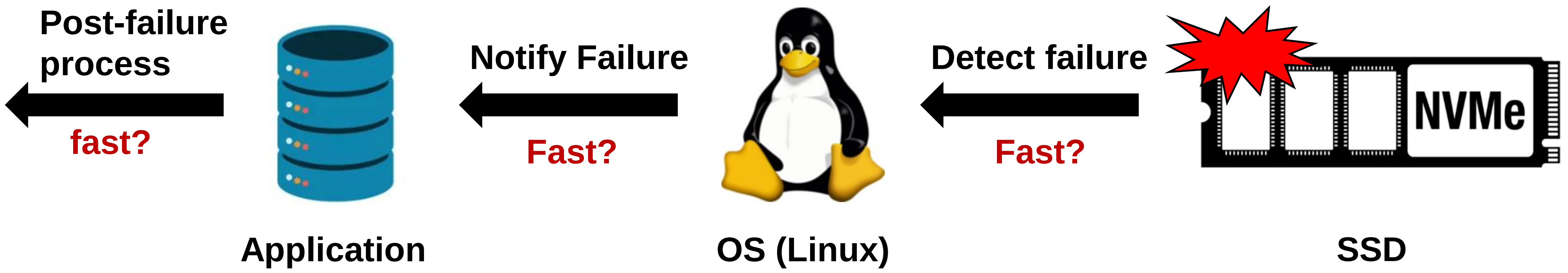
- More failure occurs
- **Failure handing is important**



# Behavior on SSD Failure

- **Fast failure notification hastens post-failure job**
  - Stop access to failed SSD, start recovery process, etc.

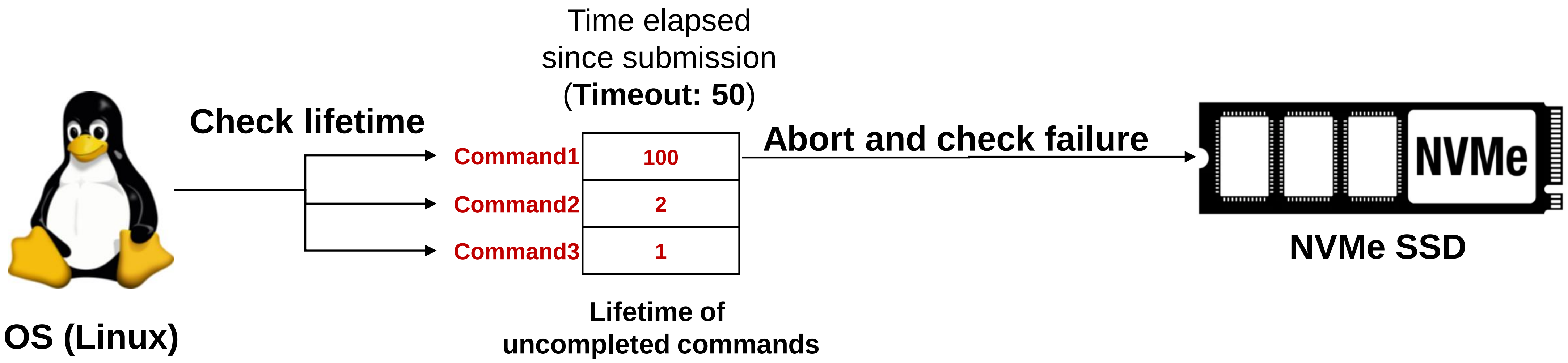
- **However,**
  - Several problems bother fast failure detection and notification
  - Delayed failure handling induces **data loss**
    - **E.g., Applications continue to perform buffered write until failure notification**



# How to Detect SSD Failure in Existing Linux

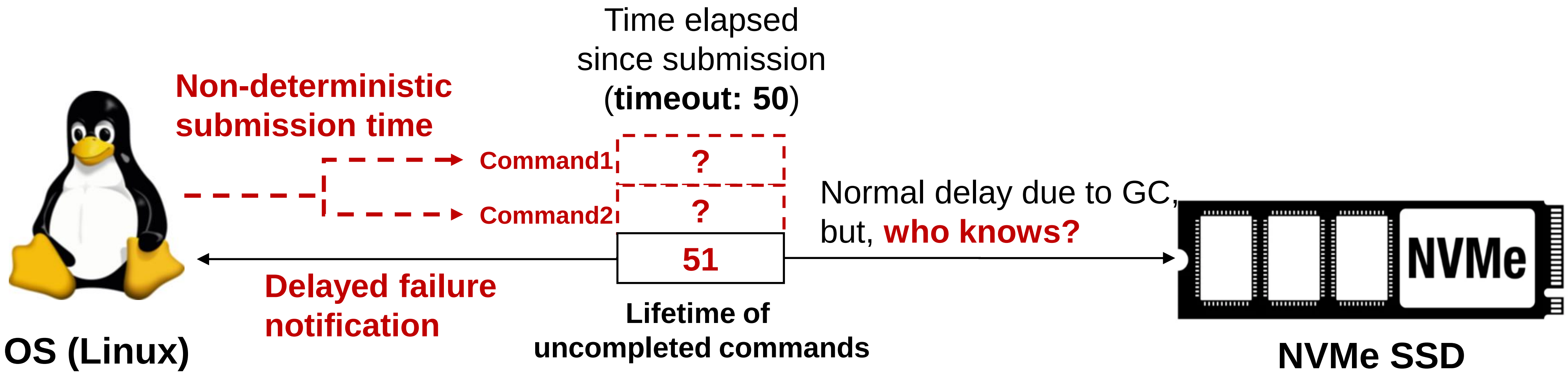
## ❖Command timeout-based detection

- Measure lifetime of uncompleted commands
- Command lifetime can be longer than command timeout
    - Linux has identified some problems with SSD
    - **Linux transfers Abort command and checks PCIe connection**



# Problem of Timeout-Based Detection

- **Accessing SSD is not deterministic**
  - E.g., **Buffered mode** can postpone I/O submission
- **Fixed timeout is not sufficient**
  - Timeout fitted to latency is required
  - E.g., **Latency fluctuates** by SSD internal operations such as garbage collection
- **Notification to application can be delayed or unavailable**



# Obstacles of Fast Failure Handling (1)

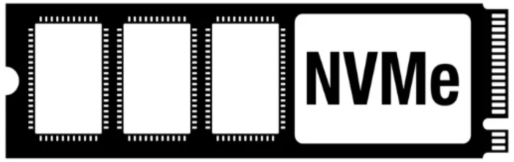
❖ **Loose-deterministic failure check**

- Delayed I/O submission delays failure checking
- **I/O submission can be postponed by buffered I/O**

OS (Linux)

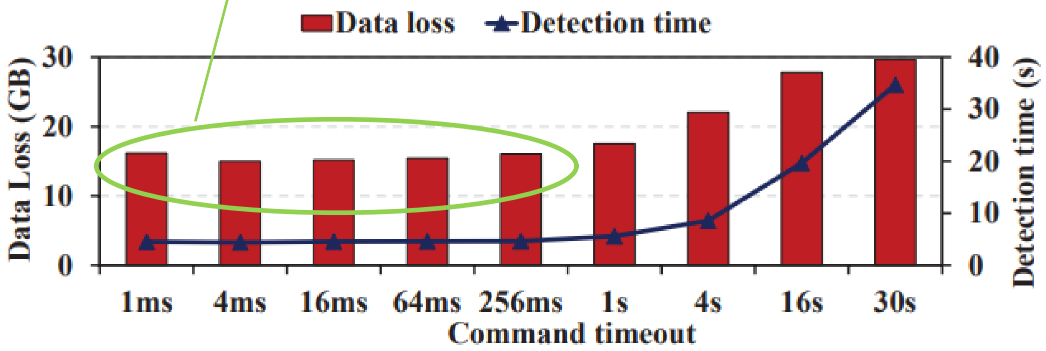
**Buffered write submission,**
**when?**

**Failure checking can be delayed**



**NVMe SSD**

**Buffered write bothers fast failure detection**



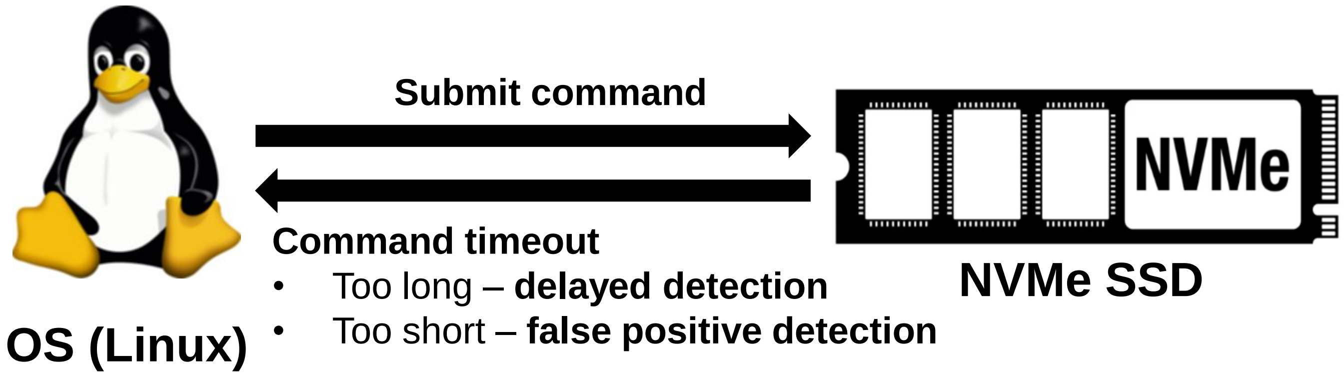

** Buffered random write by FIO, failure injected at 2 seconds*

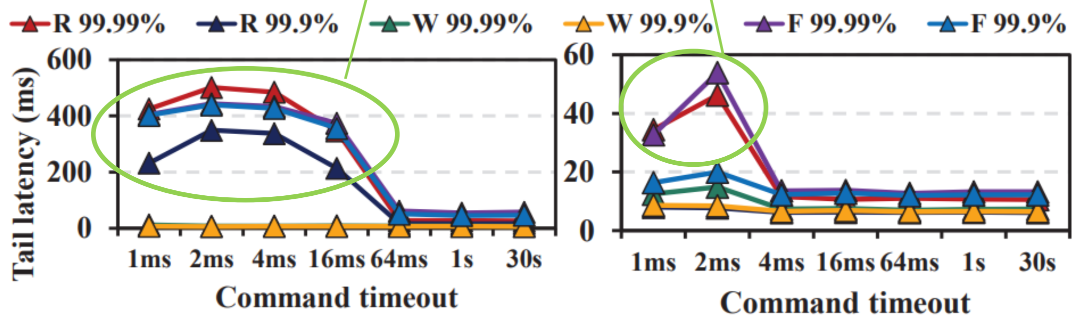
# Obstacles of Fast Failure Handling (2)

## ❖Fixed command timeout

- Suitable timeout is changed by
  - SSD models, command types, temporal business, etc.
- Fixed timeout is not appropriate solution
  - Long timeout – **delayed failure detection**
  - Short timeout – **false positive failure detection**

**False positive detection incurs overhead** (aborting command)



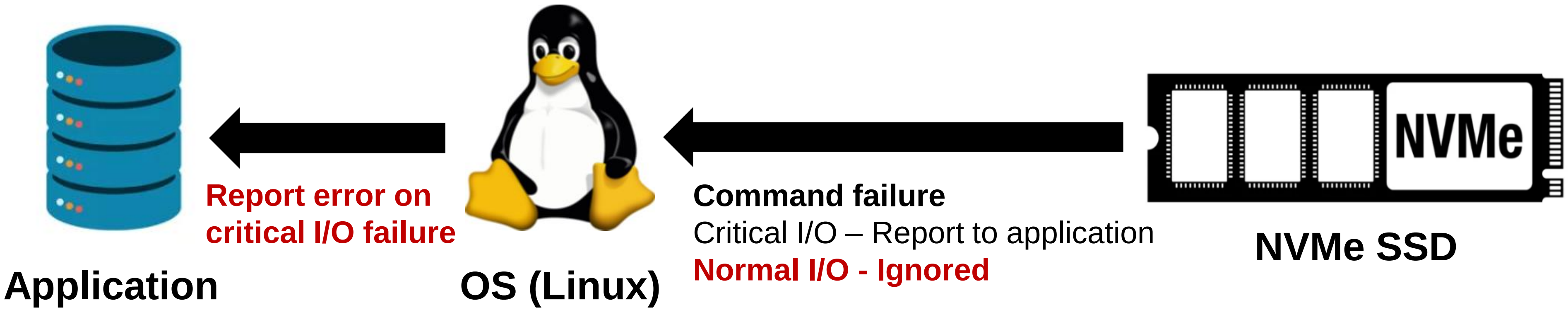

(a) Samsung 980 (SSD A) (b) Samsung PM9A3 (SSD B)

** Buffered random write/read and flush mixed pattern by FIO*

# Obstacles of Fast Failure Handling (3)

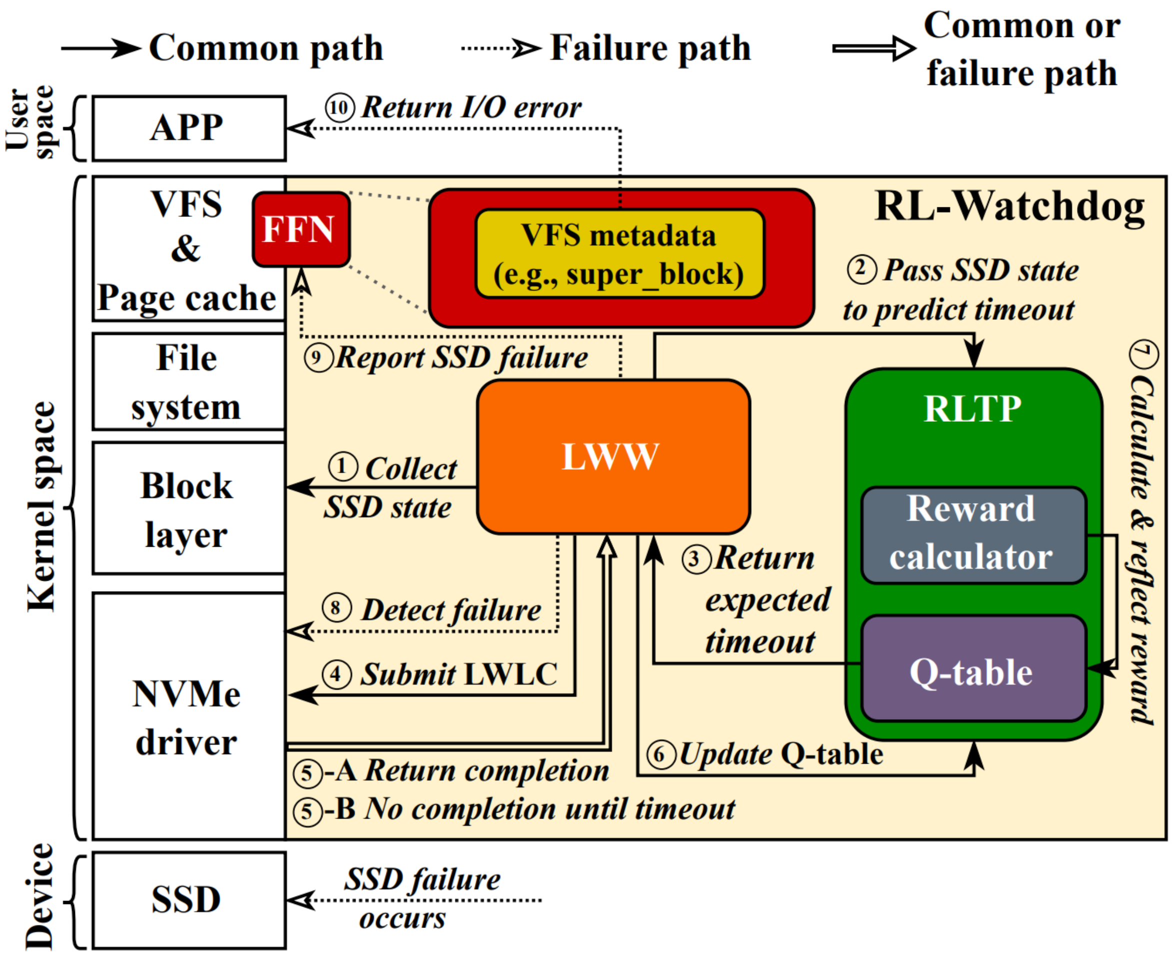
❖ **Delayed failure notification**

- Failure notification is dependent on file systems
  - E.g., **Only critical failures are notified to upper layers for buffered I/O**



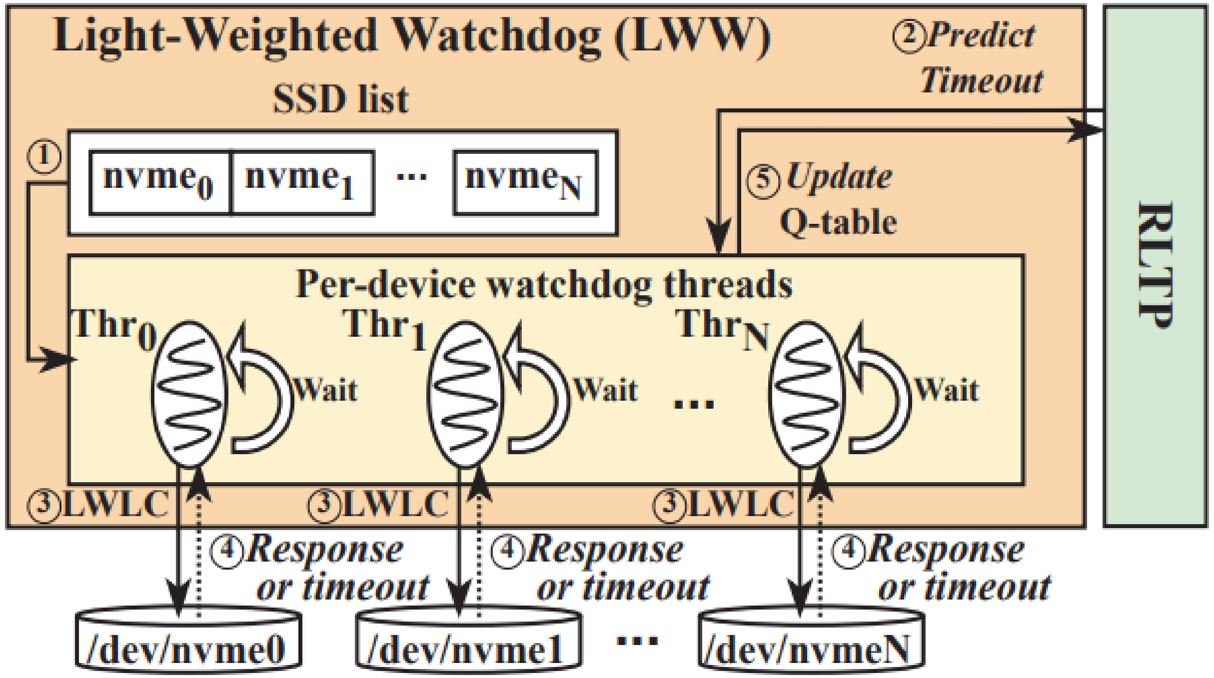
# RL-Watchdog Overview

❖ **Examine SSD liveness**
- **Light-Weighted Watchdog (LWW)**
  - Lightweight and strictly-deterministic liveness check
- **Reinforcement Learning based Timeout Predictor (RLTP)**
  - Predicting command timeout at runtime
- **Fast Failure Notification (FFN)**
  - Notifying application of SSD failure quickly



# Light-Weighted Watchdog (LWW)

- ❖ **Submit command to SSD periodically**
  - **Deterministic failure check**
  - With predicted timeout from RLTP
  - Check SSD is failed or not

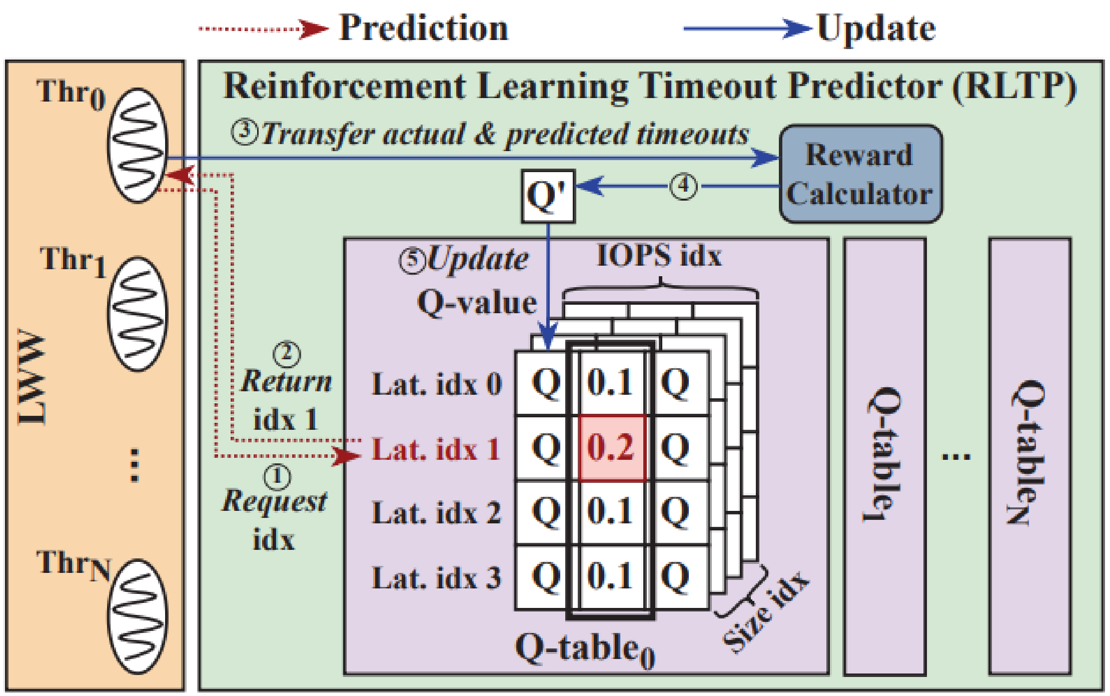



- ❖ **Light-weighted liveness-monitoring command (LWLC)**
  - Utilize reserved opcode command (**lightweight**)
  - Utilize Admin path (low interference)

# Reinforcement Learning based Timeout Predictor (RLTP)

❖ **Predict timeout based on current SSD states**
  - Learn suitable timeout online (Q-learning)
  - **Adaptive to SSD current states**

❖ **Relaxed complexity of prediction**
  - Using LWLC allows easy prediction



# Feature Selection for RL

## ❖Co-relation between features and LWLC latency

- Selected features to learn
  - In-flight I/Os
  - Write IOPS
  - Average write size

**Highly co-related features**

| Features | Video server | File server | YCSB | FIO (GC) | FFSB |
|---|---|---|---|---|---|
| In-flight I/Os | 0.06 | 0.13 | 0.63 | 0.03 | -0.005 |
| IOPS (W) | -0.23 | -0.06 | 0.34 | -0.01 | 0.023 |
| Avg. size (W) | 0.38 | -0.76 | 0.51 | -0.03 | 0.025 |
| IOPS (R) | 0.007 | -0.005 | -0.11 | -0.01 | 0.001 |
| Avg. size (R) | -0.002 | 0.002 | 0.04 | -0.01 | 0.001 |

## ❖Quantize features

- To learn quickly and efficiently
- To minimize Q-table
  - 384 Bytes per SSD

**Features to learn** / **Feature to predict**

| In-flight I/Os | Write IOPS | Avg. write Size | LWLC latency |
|---|---|---|---|
| < 13 | < Max/256 | < 8 KB | < 1 ms |
| >= 13 | < Max/16 | < 32 KB | < 4 ms |
| | >= Max/16 | < 128 KB | < 16 ms |
| | | >= 128 KB | >= 16 ms |

# Fast Failure Notification

❖ **Notify failure directly to VFS layer**
  - **Fast failure notification regardless of the policy of intermediate layers**
  - Reserve a field in VFS layer to represent SSD failure

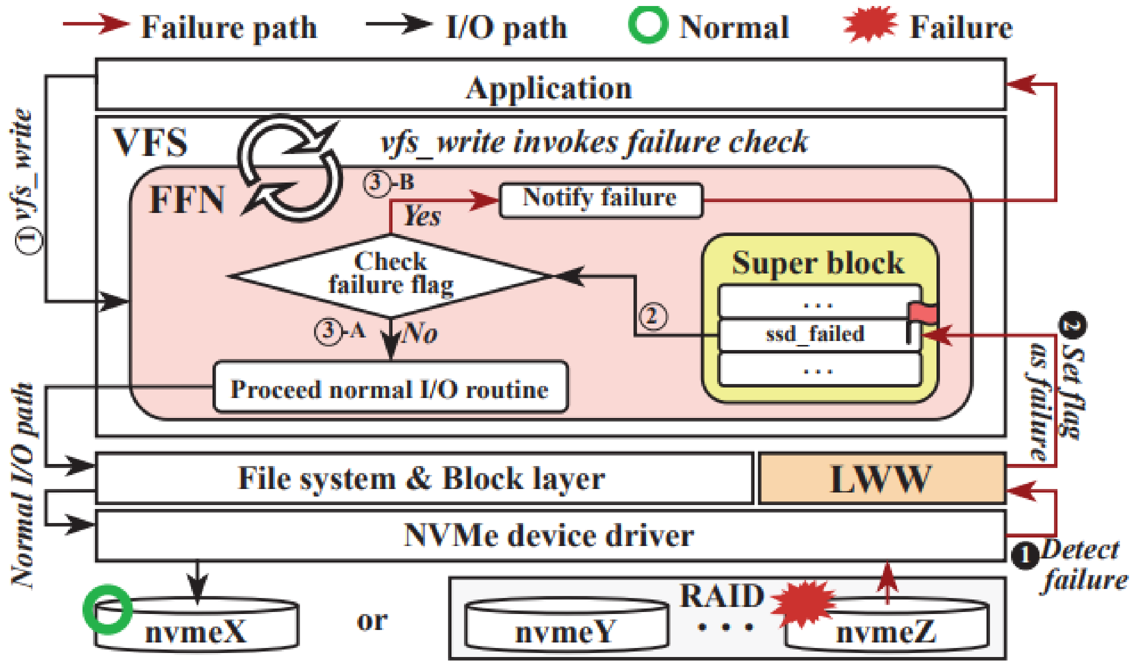



Figure 8: Procedure of FFN.

# Experiments

- **Server**
  - Xeon E5-2650 CPU (24 cores, 48 threads)
  - 160GB DRAM
  - Samsung 980, PM9A3 SSDs
  - RAID5 with 3 same SSDs
  - **Power control board**
    - Real power failure injection to SSD
    - Inject **SSD failure at 2 seconds**

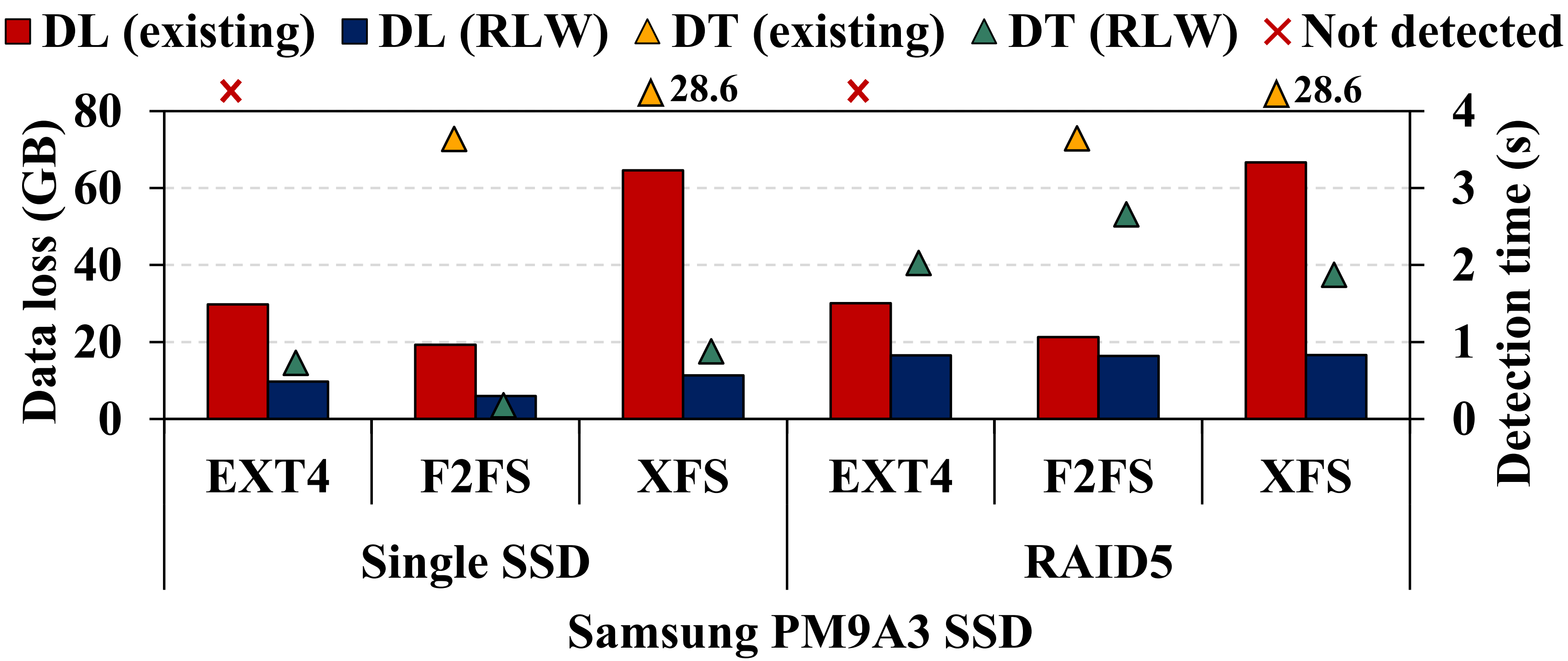

- **Workloads**
  - Buffered random write (FIO)
  - Real application (RocksDB)
    - DBBench, YCSB

- **Metrics**
  - How **much** data loss reduced?
    - Data loss (DL)
  - How **fast** failure detected?
    - Failure detection time (DT)
  - How much **accurately predict** timeout?
    - Prediction accuracy

# Buffered Writes

## ❖Data loss
- RLW reduces by up to **82.4%**

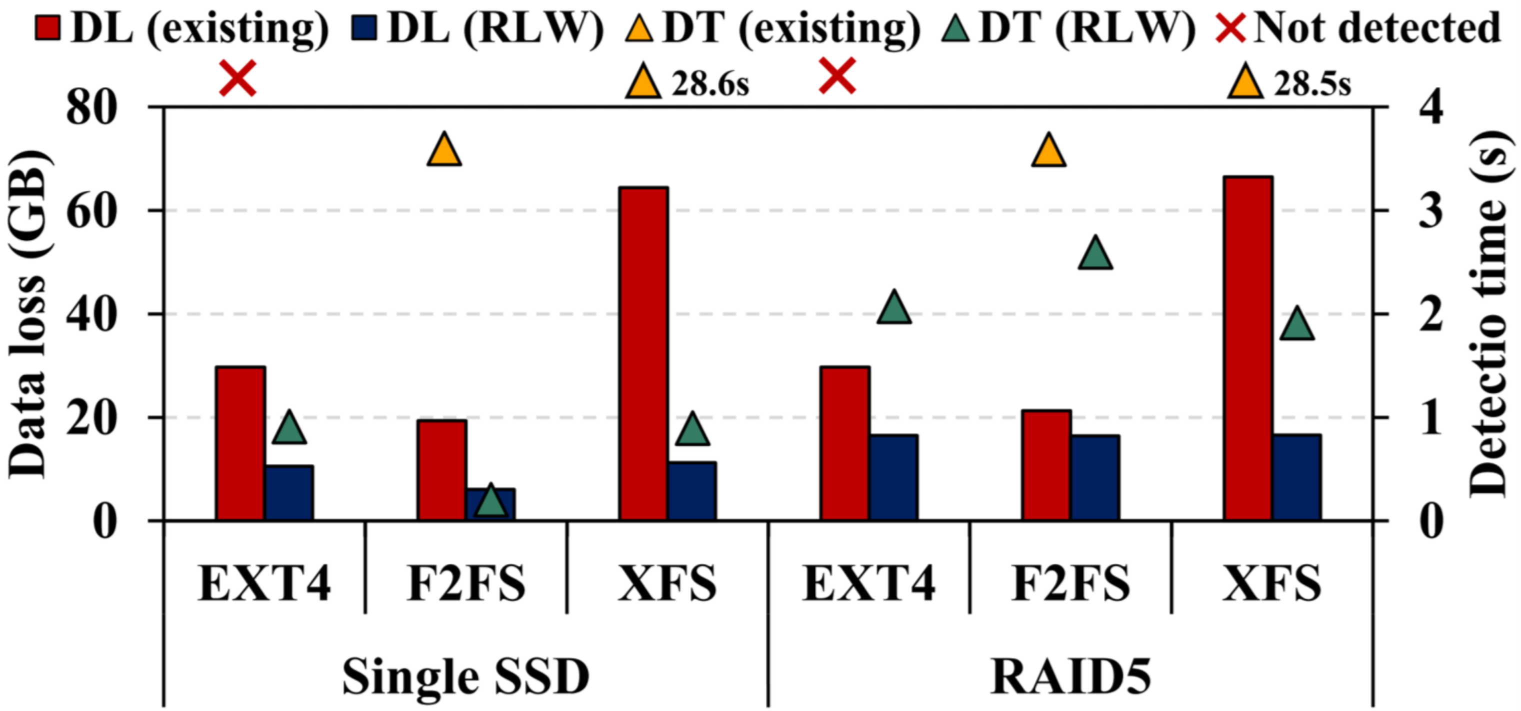
## ❖Detection time
- RLW reduces by up to **97.9%**
- Even no failure notification to application on EXT4 with existing scheme



# Different SSD Models

## ❖With different models of SSDs

- RL-watchdog effectively reduces data loss as well
- Reduce data loss and detection time, by up to **82.5% and 93.7%,** respectively

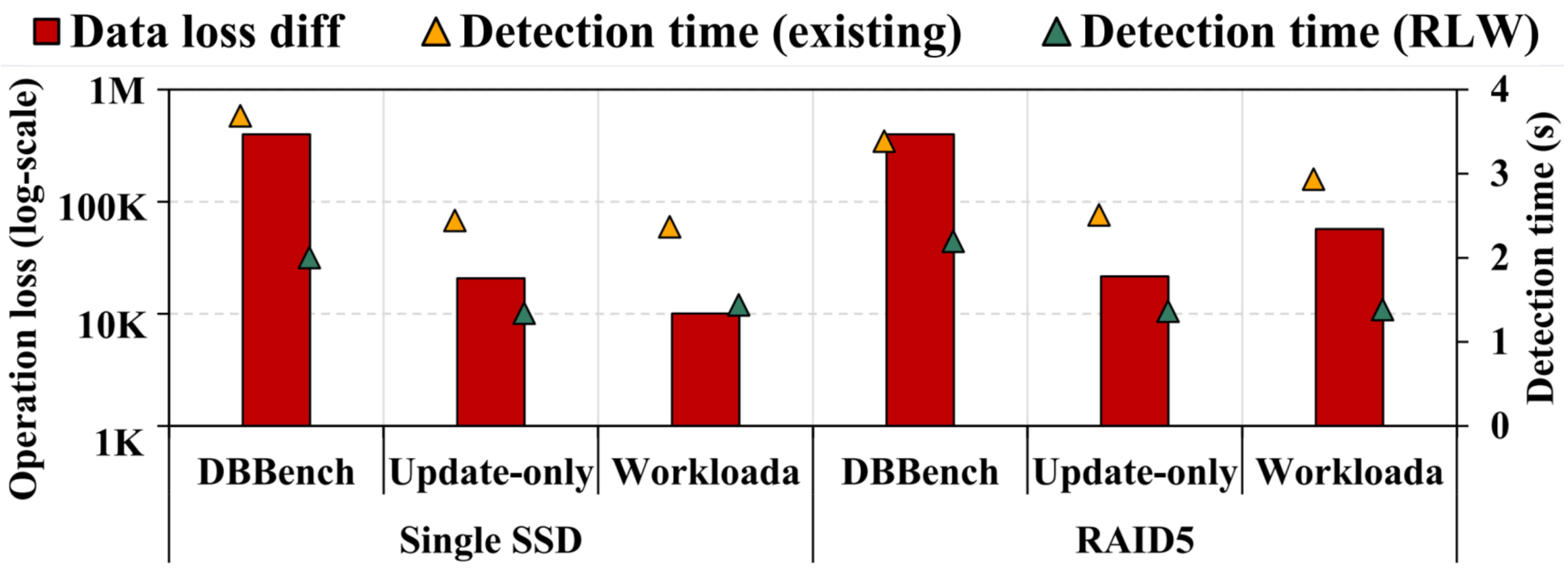



Samsung 980 SSD

# Real Application (RocksDB)

## ❖Utilize fill random (DBBench), YCSB workloads

- On every case, RLW reduces **data loss by up to 400K operations** and **detection time by up to 53%**
- RLW is effective on real application as well

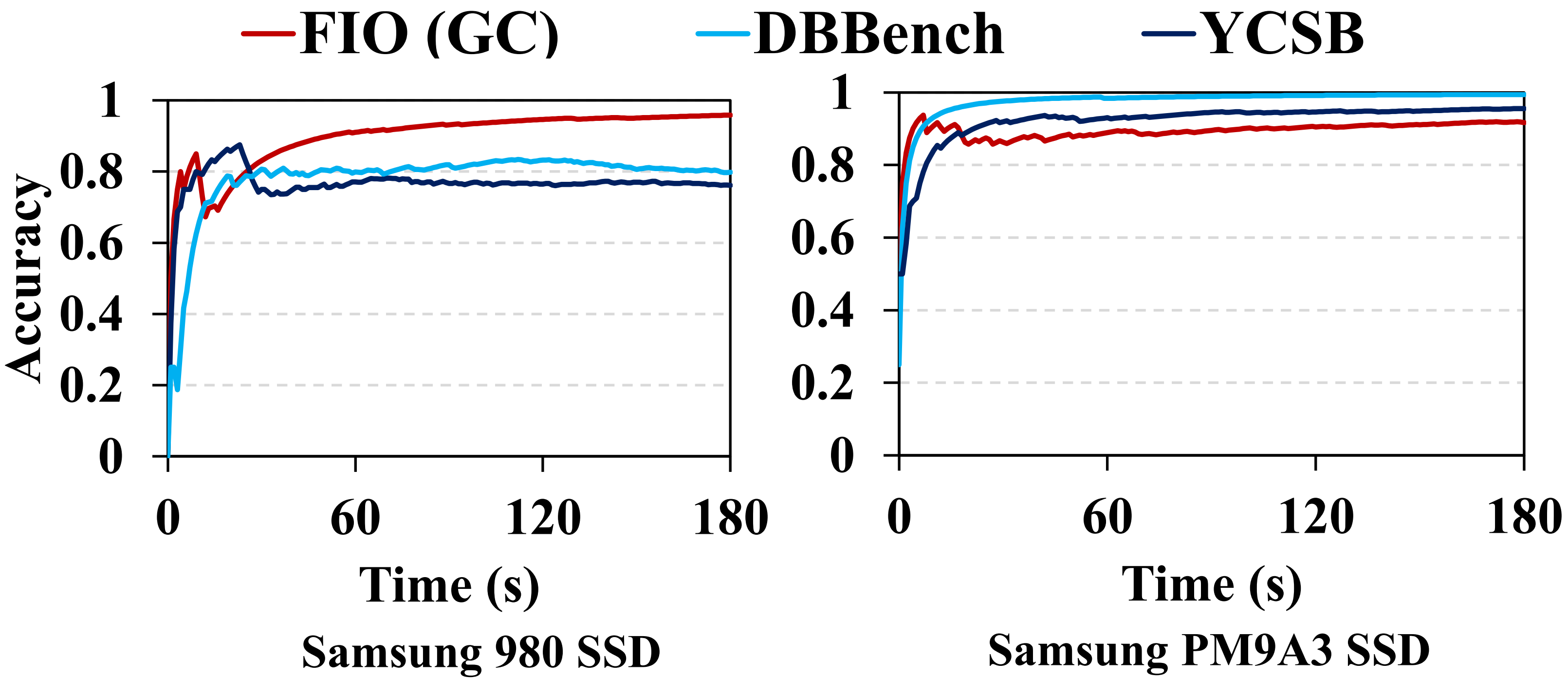



Samsung PM9A3 SSD

# Prediction Accuracy Saturation

## ❖Prediction accuracy

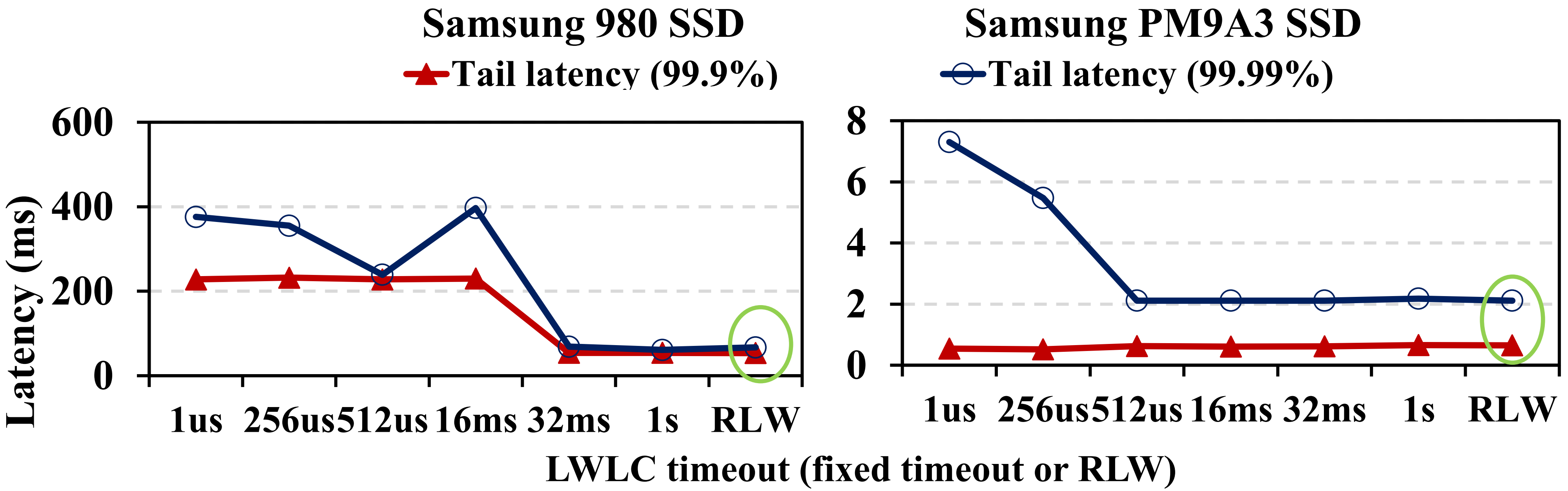
- Reaches up to **99.8%**
- Saturate at least 120 seconds
- RLTP is effective on both SSD models



# Impact of False Positive Detection

❖ **I/O tail latency**

- I/O latency increases as LWLC timeout decreases
  - Due to **false positive detection overhead**
- **No I/O interference with RL-Watchdog**
  - No false positive detection occurs in our evaluation



# Conclusion

- **RL-Watchdog examines SSD liveness or failure Quickly, Precisely, and Online to minimize application data loss**
  - Periodically monitors failures in a lightweight manner (LWW)
  - Predict command timeout precisely (RLTP)
  - Suspends storage system immediately (FFN)
  - In evaluation, RLW reduces data loss by up to 96.7% and its accuracy reaches up to 99.8%

# Thank you