# **PAT: NameNode Crash Analysis**

### **Overview**

The PAT environment NameNode crash was primarily caused by **namespace bloat** (rapid fsimage growth combined with excessive snapshots) and **intense metadata activity** that triggered repeated checkpoint cycles. This combination created severe contention on the filesystem lock, leading to instability and ultimately a crash.

### **What Happened**

* The **NameNode crashed** during a period of **heavy workload and continuous checkpointing**.
* The **edit log transaction threshold** (fs.checkpoint.txns) was exceeded multiple times in quick succession, forcing **back-to-back checkpoints**.
* Each checkpoint required the system to **load and replay a very large fsimage (~16 GB, compared to ~4.5 GB only a few months ago)**, representing a significant increase in metadata volume.
* During checkpoint operations, the **fs.lock\_queue\_length metric spiked**, showing that all client and system threads were waiting on the global filesystem lock.
* As lock contention grew, **jobs slowed, requests queued, and eventually the NameNode became unstable and crashed**.

### **Key Contributors**

1. **Large FSImage (~16 GB)**
   * Namespace growth dramatically increased checkpoint duration and memory requirements.
2. **Frequent Checkpoints**
   * High transaction rates pushed the system past the fs.checkpoint.txns threshold repeatedly, resulting in continuous checkpointing cycles.
3. **Excessive Snapshots**
   * This environment contains **many more snapshots** compared to dev/prod clusters, inflating namespace size and fsimage volume.
4. **High Metadata Activity**
   * A burst of create/delete/block allocation operations coincided with checkpoints, further driving transaction volume and lock contention.

### **Recommendations / Solutions**

**1. Reduce FSImage Size**

* Clean up stale data and unnecessary files.
* Remove or archive old datasets that unnecessarily inflate the namespace.

**2. Snapshot Governance**

* Establish strict retention policies to **limit snapshot count and lifetime**.
* Clean up older snapshots to reduce namespace and fsimage size.

**3. Checkpoint Tuning**

* While increasing fs.checkpoint.txns (e.g., to 2–3M) would reduce checkpoint frequency, this is **not advisable until fsimage size and snapshots are under control**.
* The **immediate priority** is to shrink the fsimage and reduce snapshots, ensuring checkpoints run faster and with less lock contention.