Example trace of ZOOKEEPER-4646

zk-server	S0	S1	S2
current epoch	1	1	1
last logged zxid	<1, 3>	<1, 3>	<1, 3>

Initial state. S2 is the leader.

zk-server	S0	S1	S2
current epoch	1	1	1
last logged zxid	<1, 3>	<1, 3>	<1, 4>

S2 logs a new txn <1, 4> and makes a broadcast.

zk-server	S0	S1	S2
current epoch	1	1	1
last logged zxid	<1, 3>	<1, 3>	<1, 4>

S0 & S1 crash before they receive the proposal of <1, 4>.

(Note: To ensure that majority of nodes are running at any moment, we can let S0 & S1 shutdown & restart one by one in a short time.)

S0 & S1 restart. S2 is elected leader again.
S0 & S1 DIFF sync with S2 .

zk-server	S0	S1	S2
current epoch	2	2	2
last logged zxid	<1, 3>	<1, 3>	<1, 4>

S0 & S1 send ACK-LD to S2
before their SyncRequestProcessor threads log txns to disk.

Clients of S2 have the view of <1, 4>.

zk-server	S0	S1	S2
current epoch	2	2	2
last logged zxid	<1, 3>	<1, 3>	<1, 4>

The followers S0 & S1 crash
before their SyncRequestProcessor threads persist txns to disk.
(This is extremely timing sensitive but possible!)

(Note: To ensure that majority of nodes are running at any moment, we can let S0 & S1 shutdown & restart one by one in a short time.)

Shutdown S2, and restart S0 & S1.

zk-server	S0	S1	S2
current epoch	3	3	2
last logged zxid	<1, 3>	<1, 3>	<1, 4>

After S0 & S1 finish SYNC and get into BROADCAST, clients of S0 & S1 do NOT have the view of txn <1, 4>, a violation of ZAB.

```
// In Learner.java
protected void syncWithLeader(long newLeaderZxid) throws Exception {
    ...
    synchronized (zk) {
        ...
        while (self.isRunning()) {
            ...
            switch (qp.getType()) {
                ...
                case Leader.NEWLEADER: // Getting NEWLEADER here instead of in discovery
                    // means this is Zab 1.0
                    ...

                    self.setCurrentEpoch(newEpoch);

                    ...

                    // ZOOKEEPER-3911: make sure sync the uncommitted logs before commit them (ACK NEWLEADER).
                    sock.setSoTimeout(self.tickTime * self.syncLimit);
                    self.setSyncMode(QuorumPeer.SyncMode.NONE);
                    zk.startupWithoutServing();
                    if (zk instanceof FollowerZooKeeperServer) {
                        FollowerZooKeeperServer fzk = (FollowerZooKeeperServer) zk;
                        for (PacketInFlight p : packetsNotCommitted) {
                            fzk.logRequest(p.hdr, p.rec, p.digest);
                        }
                        packetsNotCommitted.clear();
                    }

                    writePacket(new QuorumPacket(Leader.ACK, newLeaderZxid, null, null), true);
                    break;
            }
        }
    }
    ...
}
```

Add the logging requests to the queue of the SyncRequestProcessor which will process requests asynchronously.

Reply ACK-LD here.

Txn logging is processed by the SyncRequestProcessor thread asynchronously.
It is possible that, the leader receives ACK-LD and gets into BROADCAST phase before follower's SyncRequestProcessor thread starts to process txn logging.
If the follower crashes at this moment, those txns might be lost.

```
// Leader.java
void lead() throws IOException, InterruptedException {
    ...
    try {
        ...
        self.setZabState(QuorumPeer.ZabState.SYNCHRONIZATION);
        try {
            waitForNewLeaderAck(self.getId(), zk.getZxid());
        } catch (InterruptedException e) {
            ...
        }
        startZkServer();
        ...
        self.setZabState(QuorumPeer.ZabState.BROADCAST);
        ...
    } finally {
        ...
    }
}
```