Service Incident Report for the U.S. ATLAS Tier-1 Center at BNL

ATLAS file loss due to storage controller failure resulting in file system corruption.

Description.

The backend hardware RAID controller suffered a hardware problem leading to memory cache corruption which resulted in corruption of the ZFS filesystem metadata.

As confirmed by Oracle ZFS engineers, this led to corruption of labels on each of the vdevs (virtual devices) in the data pool. Importing and recovering the ZFS data pool without knowing the root MOS (Meta Object Set) location is not possible.

Impact.

There were 618,915 files lost. These were disk-resident files (T0D1) belonging to datadisk, userdisk, mcdisk, groupdisk, atlasproddisk and scratchdisk. The loss affects ATLAS managed production and user analysis.

Time line of the incident.

- June 20, 2013, at 12:32PM, a Nexsan storage controller in front of a 64 TB disk array stopped serving data and became unresponsive. Investigating the problem we discovered the controller was in a hung state. The only available option to revive the system was to restart the controller. The controller failed to boot reporting multiple ECC errors.
- At 12:50PM, the storage controller was replaced with a new spare controller and brought online. At this point the ZFS zpool on the server connected to the storage switched to faulted mode, reporting corruption at the metadata level.
- The storage which was originally on a Solaris 10.6 OS was connected to Solaris 11.1 and OpenIndiana operating systems in an attempt to roll back transactional groups (txgs) in order to salvage the data, a support case was also opened with Oracle support.
- Using the ZFS debugger an unsuccessful attempt was made to roll back txgs to a state just before the corruption using the only valid uberblock available.
- The individual LUNs were scanned entirely to find traces of valid ZFS uberblock/labels.
- June 21, a list of lost files was prepared and made available to DDM operations and ATLAS users.

Analysis.

- The ZFS label contents of each raw device array were dumped and analyzed using dd and octal dump.
- It was confirmed that there was only one valid uberblock on the labels of each LUN, furthermore, the areas of the labels where the other uberblocks should have been (the 128

Kbyte sequential areas each containing 128 uberblocks right next to another) were seemingly overwritten by random data.

- The rootbp Data Virtual Addresses from this valid uberblock were pointing to a 512-byte range that contained all zeros, and not to a valid L0 DMU Object structure, making it impossible to reconstruct the binary tree structure of ZFS.
- The corruption most likely was caused by the backend hardware controller. In essence, the failure resulted in either writes being lost (which would explain the lack of data at the root block pointer location), or writes randomly going to different blocks than what was requested (which could explain the uberblock structure corruption).
- To minimize risk of data loss in the future, regular frequent backups will be taken of the uberblock array/labels in order to locate the meta object set (MOS) to salvage the pool in case of similar corruption.