HBase Backup and Restore

Ted Yu
About myself

• Been working on HBase for over 6 years
• HBase committer / PMC
• Senior Staff Engineer at Hortonworks
HBase Backup – Why We Need It

- Database needs disaster recovery tool
- Previously users can perform snapshot
- However, execution cost may be high – flush across region servers is involved
- There was no incremental snapshot – whole dataset is captured by snapshot
HBase Backup Types

- **Full backup** – foundation for incremental backups
- **Incremental backup** – can be periodic to capture changes over time
- **Supports table level backup**
Backup Strategy

• Intra-cluster backup is appropriate for testing
Backup Strategy: Dedicated HDFS Cluster

- backup on a separate HDFS archive cluster
Backup Strategy: Cloud or a Storage Vendor

- vendor can be a public cloud provider or a storage vendor who uses a Hadoop compatible file system
Best Practices for Backup-and-Restore

• Secure a full backup image first
• Formulate a restore strategy and test it
• Define and use backup sets for groups of tables that are logical subsets of the entire dataset
• Document the backup-and-restore strategy, and ideally log information about each backup
Creating/Maintaining Backup Image

• Run the following command as hbase superuser:

  • hbase backup create
  • {{ full | incremental }}
  • {backup_root_path}
  • {[[-t tables] | [-set backup_set_name]]}
  • {[[-silent] | [-w number_of_workers] | [-b bandwidth_per_worker]]}
Using Backup Sets

• Reduces the amount of repetitive input of table names.
• “hbase backup set add” command.
• You can have multiple backup sets
• Backup set can be used in the “hbase backup create” or “hbase backup restore” commands
Restoring a Backup Image

• You can only restore on a live HBase cluster
• Run the following command as hbase superuser
• hbase restore {[-set backup_set_name] | [backup_root_path] | [backupId] | [tables]} [[table_mapping] | [-overwrite] | [-check]]

• hbase restore /tmp/backup_incremental backupId_1467823988425 mytable1,mytable2 - overwrite
Incremental backups

- Use Write Ahead Logs (WALs) to capture the data changes since the previous backup
- Log roll is executed across all RegionServers
- All the WAL files from incremental backups between the last full backup and the incremental backup are converted to HFiles
- A process similar to the DistCp tool is used to move the source backup files to the target file system
Restore

- The full backup is restored from the full backup image.
- HFileSplitter job will collect all HFile(s), split them into new region boundaries.
- HBase Bulk Load utility automatically imports the HFiles as restored data in the table.
Bulk load support

- Bulk loaded Hfiles are recorded in backup table at the end of bulk load
- During incremental backup, these Hfiles are copied to backup destination
- During restore, these Hfiles are loaded into target table
Limitations of the Backup-Restore

• Only one active backup session is supported. HBASE-16391 will introduce multiple-backup sessions support
• Both backup and restore can’t be canceled while in progress. (HBASE-15997,15998)
• Single backup destination only supported. HBASE-15476
• There is no merge for incremental images (HBASE-14135)
• Only superuser (hbase) is allowed to perform backup/restore
Credit

• Richard Ding
• Vladimir Rodionov
Thank you.