Data at Rest Encryption Test Plan

Overview

This test plan focuses on testing data at rest encryption (HDFS-6134) with the Hadoop Key Management Server (HADOOP-10433).

Platforms

We will test using the following HDFS configurations

- Vanilla
- Secure
- HA (QJM)
- HA (QJM) & secure
- Federation

Tests

Basic Functionality

Most of these cases will already exist in unit tests as part of feature development. We should look into increasing coverage where appropriate.

- Create encrypted zones and read/write data within the encrypted zones.
- Remove the encrypted zones (remove the dir).
- Verify that encrypted data adheres to the usual HDFS permissions, including ACLs.
- Verify that key rotation within a Key Store functions properly.
- Verify moving and copying data within the same encryption zone is possible, and that we cannot rename data to different encrypted zones.
- Run the basic tests using the KMS and KeyStores provided by Hadoop.

Configuration

- Specify both valid and invalid values and verify behavior.
  - Invalid values should be correctly error logged and perhaps fail-fast in some cases
- Set different ACLs in the KMS configuration and verify that they are adhered to
High Availability

- Create encryption zones and encrypted files, and then failover to the Standby NameNode and verify that the encryption zones exist and the files are encrypted
- Verify that checkpointing is successful after failing over to the Standby NameNode
- Loop failing over back and forth in between encryption zone/file modifications and verify that the changes persist
- Create encryption zones and encrypted files and then kill the Active NameNode to trigger automatic failover and verify that the encryption zones / files are consistent
- Restart entire HDFS and verify that encryption modifications persist and checkpointing works

Admin Commands and Tools

Verify that we can perform the following functions on a system with multiple encryption zones and encrypted files:

- Save the namespace
- Roll edits
- Run the OfflineEditsViewer
- Run the OfflineImageViewer
- Run Fsck specifying the path of an encryption zone and again with the root

Backup with DistCp and Snapshots

**DistCp**

- DistCp within the same encryption zone should succeed
- DistCp of encrypted data from one server to another server that has encryption enabled should succeed, as long as the target is not inside an encryption zone different from the source
- Users should not be able to DistCp data between different encryption zones
- Older clients trying to DistCp encrypted data from an encryption-enabled server should fail fast and throw an exception
- Older clients should be able to DistCp non-encrypted data from an encryption-enabled server

**Snapshots**

- Verify that users can not modify encryption properties of snapshots
- Verify that users can take snapshots of encrypted data
  - Read the contents of the encrypted snapshotted data and copy the data back into the encryption zone
- Verify that users can take snapshot diffs of snapshots of encrypted data. Note that users cannot enable encryption on a non-empty dir, so we don’t have to worry about taking snapshot diffs of an empty dir before and after encryption is enabled.

**WebHDFS and HTTPFS**

- Verify that encryption-related commands can be performed using WebHDFS and HTTPFS
- Verify that encryption-available WebHDFS clients can work with older servers where encryption is not available
- Verify that older WebHDFS clients can work with encryption-enabled servers when not accessing files inside an encryption zone.

**NFSv3**

- Verify reading and writing successfully to encrypted zones through the NFS mount.

**Fuse-DFS**

- Verify reading and writing successfully to encrypted zones through the Fuse-DFS mount.

**HFTP**

- Verify reading encrypted data successfully using the Hftp filesystem. Note that Hftp is read-only.

**Compatibility**

- Verify rolling upgrade works properly
  - Read/write to HDFS during the rolling upgrade
- Older Hadoop clients in the same major version should be able to read/write to encryption-enabled servers as long as the data accessed is not within an encryption zone
  - Accessing encrypted data from an old client should fail fast and throw a reasonable exception and not return encrypted data
- Encryption-enabled clients should be able to read/write to older servers
- WebHDFS and DistCp compatibility tests are mentioned earlier in this document.

**Hadoop Archives**

- Create a hadoop archive that includes encrypted contents
- Access the encrypted content using the *har* filesystem
Short Circuit Reads

- Enable short circuit reads and read encrypted data successfully.

Error Testing

- Kill the KMS server and then try to R/W encrypted files and create encryption zones.
  - Are the errors descriptive enough for debugging failures?
  - Are the logs descriptive enough?
- Give some invalid values to CLI commands and check that they return a reasonable error
  - e.g. Give too many arguments and check we don’t get an ArrayIndexOutOfBoundsException, invalid path on deletion/creation

ViewFs

- Run the Encryption related Java APIs and FsShell commands using a ViewFs configuration
- Verify that the encryption commands are routed to the correct NameNode

S3

Using both FsShell -cp and DistCp:

- Attempt to copy encrypted data in HDFS to S3
- Copy data from S3 to an encrypted zone

Quotas

- Verify that quotas are enforced within encryption zones
  - The encryption XAttrs do not count against disk space quota usage
  - Run the count quota commands encrypted dir/files and verify numbers are correct

Trash

- Remove encrypted data into the trash successfully or give an appropriate error message
  - Copy that data back to the encryption zone
  - Expunge that data from the trash

Web UI
- Verify the new web UIs for correctness and completeness (KMS server has a new web UI)
- Verify HDFS web UI shows encrypted zones and data properly (use the file browser)

**Memory Caching**

Operation with HDFS caching is *not a required exit criteria* for the first release. However, we should still try to enable/disable caching on data within encrypted zones to see what happens.

**Downstream projects**

- Run MapReduce R/W jobs on encrypted data.
  - For example, TeraGen data within an encryption zone, and then run TeraSort and TeraValidate.

Other downstream projects such as Hive and Pig will need to test integration with Data at Rest Encryption, similar to how they tested integration with ACLs.

**Review documentation**

- Review the upstream Hadoop documentation for correctness and completeness

**Stress/Performance**

- Proper operation of the system with multiple zones at scale. This should include stress testing of (e.g.) K’s of zones.