Introduction

The UserGroupInformation(UGI) class of Hadoop security exposes different methods to get the current user/login user. In this document, we will discuss in terms of replication, all UGI and Kerberos login specific tasks that need to be performed for replication to work seamlessly in a kerberized cluster.

Login using Keytab

The Hive process starts with the hive keytab. So the loginUser(getLoginUser API of UGI class) will always be hive/HOST. It doesn't depend on which client you use to login to Hive. All connections from Hive will always use the hive keytab(exceptions proxy users will discuss below). The current user(getCurrentUser) is also hive, unless a proxy user is created. This is done in case of doAs=true, where a proxy user is created and the current user becomes the proxy user.
RetryingMetastoreClient does a re login through keytab to make sure if the TGT is expired, we refresh the tokens. So any client calls which use the RetryingMetastoreClient needn't worry about re login.
Replication also does atlas and ranger replication along with Hive. So there may be scenarios where we don't connect to Metastore or use the RetryingMetastoreClient. Also since the retry can extend upto 24 hrs or more, there are chances where the tokens may expire. So replication code needs to take care of re-login, before doing a retry. The re-login should be done with the login user(which is hive).

The method to do a relogin is present in SecurityUtil.

```
public static void reloginExpiringKeytabUser() throws MetaException {
 if(!UserGroupInformation.isSecurityEnabled()){
   return;
 }
 try {
   UserGroupInformation ugi = UserGroupInformation.getLoginUser();
   if(ugi.isFromKeytab()){
     ugi.checkTGTAndReloginFromKeytab();
   }
 } catch (IOException e) {
   String msg = "Error doing relogin using keytab " + e.getMessage();
   LOG.error(msg, e);
   throw new MetaException(msg);
 }
}
```

DoAs Privilege Action

Driver doesn't do a DoAs privilege action. So individual calls need to take care of this.

HTTP Calls : All HTTP calls need to do a doAs privilege action using the UGI information. If that is not done, the HTTP client will not get the UGI information and the connection will fail. We can either use the current user or the login user to do the privilege action. The login user will always be hive. The current user may be different if a proxy user is used or doAs is supported. However Replication does not support doAs, so we will use the login user to do the privilege action for now. Once we start supporting doAs in replication, we can modify it to use the current user. doAs support in replication would mean modifying all FS calls also to use the proxy user.

```
UserGroupInformation.getLoginUser().doAs(new PrivilegedAction<Object>() {
 @Override
 public Object run() {
   //Make HTTP Calls here
 }
});
```

HDFS Calls : doAs privilege action is needed only for using the proxy user. Otherwise the UGI information is picked up directly. Once replication starts supporting doAs, all HDFS calls would need to be moved inside a privilege action which will be executed as the proxy user.

```
UserGroupInformation.createProxyUser("<proxy user name>", UserGroupInformation.getLoginUser()).doAs(new
PrivilegedAction<Object>() {
 @Override
 public Object run() {
   //Make HDFS Calls here for using the proxy user
 }
});
```