Secure Impersonation using UserGroupInformation.doAs

Table of contents

1. Introduction

This document describes how a superuser can submit jobs or access hdfs on behalf of another user in a secured way.

2. Use Case

The code example described in the next section is applicable for the following use case.

A superuser oozie wants to submit job and access hdfs on behalf of a user joe. The superuser has kerberos credentials but user joe doesn't have any. The tasks are required to run as user joe and any file accesses on namenode are required to be done as user joe. It is required that user joe can connect to the namenode or job tracker on a connection authenticated with oozie's kerberos credentials. In other words oozie is impersonating the user joe.

3. Code example

In this example oozie's kerberos credentials are used for login and a proxy user ugi object is created for joe. The operations are performed within the doAs method of this proxy user ugi object.

```
            ...
            UserGroupInformation ugi =
                    UserGroupInformation.createProxyUser(user,
UserGroupInformation.getLoginUser());
            ugi.doAs(new PrivilegedExceptionAction<Void>() {
              public Void run() throws Exception {
                //Submit a job
                JobClient jc = new JobClient(conf);
                jc.submitJob(conf);
                //OR access hdfs
                FileSystem fs = FileSystem.get(conf);
                fs.mkdir(someFilePath);
              }
            }
```

4. Configurations

The superuser must be configured on namenode and jobtracker to be allowed to impersonate another user. Following configurations are required.

```
        <property>
          <name>hadoop.proxyuser.oozie.groups</name>
          <value>group1,group2</value>
```

```
                  <description>Allow the superuser oozie to impersonate any
members of the group group1 and group2</description>
             </property>
             <property>
               <name>hadoop.proxyuser.oozie.hosts</name>
               <value>host1,host2</value>
               <description>The superuser can connect only from host1 and
host2 to impersonate a user</description>
             </property>
```

If these configurations are not present, impersonation will not be allowed and connection will fail.

If more lax security is preferred, the wildcard value * may be used to allow impersonation from any host or of any user.

5. Caveats

The superuser must have kerberos credentials to be able to impersonate another user. It cannot use delegation tokens for this feature. It would be wrong if superuser adds its own delegation token to the proxy user ugi, as it will allow the proxy user to connect to the service with the privileges of the superuser.

However, if the superuser does want to give a delegation token to joe, it must first impersonate joe and get a delegation token for joe, in the same way as the code example above, and add it to the ugi of joe. In this way the delegation token will have the owner as joe.