Penn service principals from kerberos and web service authentication

We have a table (this is oracle, but you can easily translate for your database):

(note, assumes you have a hibernate_sequence Oracle sequence... you could adjust that to do something else or uuid if you like...

```
CREATE TABLE SERVICE_PRINCIPALS
(
  PRINCIPAL_NAME     VARCHAR2(512 CHAR),
  ID                 INTEGER,
  LAST_UPDATED       TIMESTAMP(6),
  REASON             VARCHAR2(4000 CHAR)
);

COMMENT ON COLUMN SERVICE_PRINCIPALS.REASON IS 'reason for adding this principal';




CREATE UNIQUE INDEX SERVICE_PRINCIPALS_PK ON SERVICE_PRINCIPALS
(PRINCIPAL_NAME);


ALTER TABLE SERVICE_PRINCIPALS ADD (
  CONSTRAINT SERVICE_PRINCIPALS_PK
  PRIMARY KEY
  (PRINCIPAL_NAME)
  USING INDEX SERVICE_PRINCIPALS_PK);
```

Then we have a subject source (subjectproperties not sources.xml):

```
########################################
## Configuration for source id: servPrinc
## Source configName: servPrinc
########################################
subjectApi.source.servPrinc.id = servPrinc

# this is a friendly name for the source
subjectApi.source.servPrinc.name = Kerberos service principals

# type is not used all that much.  Can have multiple types, comma separate.  Can be person, group, application
subjectApi.source.servPrinc.types = application

# the adapter class implements the interface: edu.internet2.middleware.subject.Source
# adapter class must extend: edu.internet2.middleware.subject.provider.BaseSourceAdapter
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter2  :  if doing JDBC this should be used if
possible.  All subject da
ta in one table/view.
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter   :  oldest JDBC source.  Put freeform queries
in here
# edu.internet2.middleware.grouper.subj.GrouperJndiSourceAdapter   :  used for LDAP
subjectApi.source.servPrinc.adapterClass = edu.internet2.middleware.subject.provider.JDBCSourceAdapter

subjectApi.source.servPrinc.param.jdbcConnectionProvider.value = edu.internet2.middleware.grouper.subj.
GrouperJdbcConnectionProvider

subjectApi.source.servPrinc.param.maxPageSize.value = 100

# ldap attribute which is the subject id.  e.g. exampleEduRegID   Each subject has one and only one subject
id.  Generally it is opa
que and permanent.
subjectApi.source.servPrinc.param.SubjectID_AttributeType.value = loginid

# attribute which is the subject name
subjectApi.source.servPrinc.param.Name_AttributeType.value = name
```

```
# attribute which is the subject description
subjectApi.source.servPrinc.param.Description_AttributeType.value = description

# the 1st sort attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 sort attributes
subjectApi.source.servPrinc.param.sortAttribute0.value = loginid

# the 1st search attribute for lists on screen that are derived from member table (e.g. search for member in
group)
# you can have up to 5 search attributes
subjectApi.source.servPrinc.param.searchAttribute0.value = loginid

subjectApi.source.servPrinc.param.useInClauseForIdAndIdentifier.value = true

subjectApi.source.servPrinc.param.identifierAttributes.value = loginid

#searchSubject: find a subject by ID.  ID is generally an opaque and permanent identifier, e.g. 12345678.
#  Each subject has one and only on ID.  Returns one result when searching for one ID.
subjectApi.source.servPrinc.search.searchSubject.param.numParameters.value = 1

# sql is the sql to search for the subject by id should use an {inclause}
subjectApi.source.servPrinc.search.searchSubject.param.sql.value = select    principal_name as name,
principal_name as loginid,   principal_name as description from    service_principals where     {inclause}

# inclause allows searching by subject for multiple ids or identifiers in one query, must have {inclause} in
the sql query,
#    this will be subsituted to in clause with the following.  Should use a question mark ? for bind variable
subjectApi.source.servPrinc.search.searchSubject.param.inclause.value = principal_name = ?

#searchSubjectByIdentifier: find a subject by identifier.  Identifier is anything that uniquely
#  identifies the user, e.g. jsmith or jsmith@institution.edu.
#  Subjects can have multiple identifiers.  Note: it is nice to have if identifiers are unique
#  even across sources.  Returns one result when searching for one identifier.
subjectApi.source.servPrinc.search.searchSubjectByIdentifier.param.numParameters.value = 1

# sql is the sql to search for the subject by identifier should use an {inclause}
subjectApi.source.servPrinc.search.searchSubjectByIdentifier.param.sql.value = select    principal_name as
name,    principal_name as loginid,    principal_name as description from    service_principals where
{inclause}

# inclause allows searching by subject for multiple ids or identifiers in one query, must have {inclause} in
the sql query,
#    this will be subsituted to in clause with the following.  Should use a question mark ? for bind variable
subjectApi.source.servPrinc.search.searchSubjectByIdentifier.param.inclause.value = principal_name = ?

#   search: find subjects by free form search.  Returns multiple results.
subjectApi.source.servPrinc.search.search.param.numParameters.value = 1

# sql is the sql to search for the subject free-form search.  user question marks for bind variables
subjectApi.source.servPrinc.search.search.param.sql.value = select    principal_name as name,    principal_name
as loginid,    principal_name as description from    service_principals where    (lower(principal_name) like
lower(concat('%',concat(?,'%'))))
```

Then we have a subject source (sources.xml (if you havent converted to subject.properties yet), not subject.properties):

```
  <!-- Service Principal Subject Resolver -->
 <source adapterClass="edu.internet2.middleware.subject.provider.JDBCSourceAdapter">
    <id>servPrinc</id>
    <name>Kerberos service principals</name>
     <type>application</type>
     <init-param>
       <param-name>jdbcConnectionProvider</param-name>
       <param-value>edu.internet2.middleware.grouper.subj.GrouperJdbcConnectionProvider</param-value>
     </init-param>
```

```
    <!-- on a findPage() this is the most results returned -->
    <init-param>
      <param-name>maxPageSize</param-name>
      <param-value>100</param-value>
    </init-param>
      <init-param>
       <param-name>SubjectID_AttributeType</param-name>
       <param-value>loginid</param-value>
     </init-param>
     <init-param>
       <param-name>Name_AttributeType</param-name>
       <param-value>name</param-value>
     </init-param>
     <init-param>
       <param-name>Description_AttributeType</param-name>
       <param-value>description</param-value>
     </init-param>
     <!-- init-param>
       <param-name>maxResults</param-name>
       <param-value>1000</param-value>
     </init-param -->

     <init-param>
       <param-name>sortAttribute0</param-name>
       <param-value>loginid</param-value>
     </init-param>
     <init-param>
       <param-name>searchAttribute0</param-name>
       <param-value>loginid</param-value>
     </init-param>
      <!-- if you are going to use the inclause attribute
        on the search to make the queries batchable when searching
        by id or identifier -->
      <init-param>
        <param-name>useInClauseForIdAndIdentifier</param-name>
        <param-value>true</param-value>
      </init-param>

      <!-- comma separate the identifiers for this row, this is for the findByIdentifiers if using an in clause
-->
      <init-param>
        <param-name>identifierAttributes</param-name>
        <param-value>loginid</param-value>
      </init-param>
     <search>
         <searchType>searchSubject</searchType>
      <param>
          <param-name>numParameters</param-name>
          <param-value>1</param-value>
        </param>
         <param>
             <param-name>sql</param-name>
             <param-value>
select
   principal_name as name,
   principal_name as loginid,
   principal_name as description
from
   service_principals
where
    {inclause}
             </param-value>
          </param>
          <param>
              <param-name>inclause</param-name>
              <param-value>
 principal_name = ?
             </param-value>
          </param>
     </search>
```

```
     <search>
         <searchType>searchSubjectByIdentifier</searchType>
      <param>
          <param-name>numParameters</param-name>
          <param-value>1</param-value>
        </param>
         <param>
             <param-name>sql</param-name>
             <param-value>
select
   principal_name as name,
   principal_name as loginid,
   principal_name as description
from
   service_principals
where
    {inclause}
             </param-value>
          </param>
          <param>
              <param-name>inclause</param-name>
              <param-value>
 principal_name = ?
             </param-value>
         </param>
     </search>
     <search>
        <searchType>search</searchType>
     <param>
          <param-name>numParameters</param-name>
          <param-value>1</param-value>
        </param>
         <param>
             <param-name>sql</param-name>
             <param-value>
select
   principal_name as name,
   principal_name as loginid,
   principal_name as description
from
   service_principals
where
   (lower(principal_name) like lower(concat('%',concat(?,'%'))))
             </param-value>
          </param>
     </search>
   </source>
```

Here is a GSH script (oracle) to add a new one:

```
edit kerb and reason...


[appadmin@fastprod-mgmt-01 bin]$ more createKerbTest.gsh
grouperSession = GrouperSession.startRootSession();
kerb = "test_kerb/school.edu";
reason = "test kerb";
sqlRun("insert into service_principals (principal_name, id, last_updated, reason) values ('" + kerb + "',
hibernate_sequence.nextval, systimestamp, '" + reason + "')");
addMember("school:etc:ldapUsers", kerb);
addMember("school:etc:webServiceClientUsers", kerb);
[appadmin@fastprod-mgmt-01 bin]$ ./gsh createKerbTest.gsh
```

Heres a GSH script (postgres) to add a new one:

```
grouperSession = GrouperSession.startRootSession();
kerb = "test_kerb2/school.edu";
reason = "test kerb2 for john smith in dept a";
sqlRun("insert into service_principals (principal_name, id, last_updated, reason) values ('" + kerb + "',
(select max(id)+1 from service_principals), CURRENT_TIMESTAMP, '" + reason + "')");
addMember("school:etc:ldapUsers", kerb);
addMember("school:etc:webServiceClientUsers", kerb);
```